TR-34
1971

Water Resource System Optimization by Geometric Programming

W.L. Meier
C.S. Shih
D.J. Wray

Texas Water Resources Institute

Texas A&M University

RESEARCH PROJECT TECHNICAL COMPLETION REPORT

Project Number A-011-TEX

September 1969 - August 1971

Agreement Number
14-01-0001-1864
14-31-0001-3044
14-31-0001-3244

WATER RESOURCE SYSTEM OPTIMIZATION BY GEOMETRIC PROGRAMMING

Principal Investigators

W. L. Meier, Jr.

C. S. Shih

Duane J. Wray

The work upon which this publication is based was supported in part by funds provided by the United States Department of the Interior, Office of Water Resources Research, as authorized under the Water Resources Research Act of 1964, P.L. 88-379.

Technical Report No. 34
Water Resources Institute
Texas A&M University

February 1971

PREFACE

The research described in this report was supported by the U. S. Department of the Interior, Office of Water Resources Research and the Texas A&M University Water Resources Institute. The aim of this research was to study the applicability and usefulness of geometric programming in water resource planning and design activities. In order to study the use of geometric programming in practical problem solving, a computer algorithm was developed. A variety of water resources problems were solved using the algorithm.

The authors believe that there is a significant need for techniques which will permit the optimum design and operation of either proposed or existing water filtration systems. It was also believed that, because the cost models and constraints for water filtration design problems are generalized polynomials, geometric programming would be a good optimization technique to use. Because water filtration problems have not been the subject of optimization analyses in the past to any significant degree, some time was spent in developing cost models and formulating the problem. This effort and the optimization analysis combining linear and geometric programming was of such interest that it is described rather extensively in the report. The report includes a tutorial explanation of geometric programming, a description of the computer algorithm used, the description of the water filtration cost model, and an explanation of the use of geometric programming in an application.

Appreciation is expressed to the numerous colleagues who assisted in the satisfactory completion of this research. Thanks are due to Mr. Thomas H. Edwards for his help during the initial phases of the research. The assistance and support of Dr. J. R. Runkles and the staff of the Water Resources Institute at Texas A&M University is gratefully acknowledged. Special thanks are due to Mrs. Jane Teske who spent long hours in typing the manuscript.

TABLE OF CONTENTS

		Page
	PREFACE	ii
	TABLE OF CONTENTS	iv
	LIST OF TABLES	vi
	LIST OF FIGURES	vii
Chapter		
I	INTRODUCTION	1
	Optimization in Water Resources Analyses	1
	Mathematical Programming	2
	Development of Geometric Programming	3
II	GEOMETRIC PROGRAMMING	6
	Posynomial Functions	6
	Optimizing Posynomials	7
	Constrained Optima	12
	Degrees of Difficulty	13
	Computer Algorithm	16
	Generalized Polynomials	17
III	COMPUTER ALGORITHM	22
	Generalized Notation	22
	Main Program	23
	Subroutine MAT	25
	Subroutine LETTO	25
	Subroutine VAL	29
	Other Subroutines	29
	Input	30
	Output	32
	Error Messages	33
IV	WATER FILTRATION THEORY	34
	The Filtering Process	34
	Physical Removal Mechanisms	34
	Chemical Removal Mechanisms	37
	Empirical Studies	38
	Multimedia Filters	40

Chapter		Page
V	FILTER COST MODEL	45
	Reducing the Problem	45
	Objective Function	47
	Constraints	49
	An Optimum Solution	56
VI	A SAMPLE PROBLEM	57
	Constant Values	57
	Model for Sample Problem	59
	Geometric Programming Solution	61
	Linear Programming Solution	62
	Summary of Results	63
VII	CONCLUSIONS AND RECOMMENDATIONS	66
	Filter Cost Model	66
	Geometric Programming Optimization	66
	Combining Optimization Techniques	67
	Future Extensions	68
	LIST OF REFERENCES	69
	APPENDIX	
	A Program Listing	74
	B Example of Output	108

LIST OF TABLES

Table		Page
6.1	OUTPUT COMPARISONS	65

LIST OF FIGURES

Figure		Page
3.1	PROGRAM GPS	24
3.2	SUBROUTINE MAT FLOW DIAGRAM	26
4.1	PERFORMANCE CURVE 1 FOR A SYNTHETIC MUDDY WATER	41
4.2	PERFORMANCE CURVE 2 FOR A SYNTHETIC MUDDY WATER	42
4.3	COMPARISON OF FILTER MEDIA.	43
5.1	MULTIMEDIA FILTER	46

CHAPTER I

INTRODUCTION

Optimization in Water Resources Analyses

Water resources planners and systems analysts are continually confronted with many complex optimization problems. Two major factors contribute to this problem. Firstly, mathematical modeling and system description capabilities in water resources have increased considerably. Secondly, planning and design analyses have become more complicated because of a desire by planners to represent more completely the problem to be solved.

In a recent appraisal of Federal Water Resources Research activities, research concerned with water resource planning methodology was labeled as the "most promising area of research" [22]. In an effort to meet the Congressional decree [54] for "optimum" water resources planning, the literature is filled with illustrations of the applicability and usefulness of optimization methods in water resources analyses [38]. Linear programming [53], dynamic programming [41,40], and nonlinear programming [19] have been shown to be useful in water resource optimization problems. These techniques have been and are being coupled with mathematical modeling and system simulation to produce practically useful planning algorithms.

The purpose of this report is to describe a new optimization technique which can be extremely useful in solving water resources optimization problems. This new and potentially powerful technique

is called *geometric programming*. It is one of a class of mathematical programming techniques. Mathematical programming discussed extensively elsewhere [56,23] refers to a class of optimization problems dealing with the allocation of limited resources. Problems are described with an objective function which measures system effectiveness and constraints which describe limitations on the solution. If either the objective function or constraints are nonlinear, a nonlinear programming problem results.

A variety of algorithms are available for solving nonlinear programming problems. However, when the constraints are nonlinear, the objective function is higher than second degree or the problem of high dimensionality; most of the available techniques are not computationally efficient. Little previous research has been aimed at making geometric programming of use to water resource analysts. The purpose of this research was to explore ways of making geometric programming more useful to water resource systems analysts. Specifically, this research has as its objectives the development of a computer algorithm for solving geometric programming problems and the specific application of this algorithm in the solution of a water renovation and treatment process.

Mathematical Programming

A general statement of the mathematical programming problem is

$$\begin{aligned} &\text{maximize/minimize:} && y = g(\underset{\sim}{x}) \\ &\text{subject to:} && g_i(\underset{\sim}{x}) \gtreqless b_i \quad i=1,..,N, \\ & && \underset{\sim}{x} \geq 0. \end{aligned} \tag{1.1}$$

A maximization problem is equivalent to the minimization of the negative of the objective function shown above. The forms of $g(\underset{\sim}{x})$ are important in determining the method used to solve the problem. For example if the $g(\underset{\sim}{x})$ are all linear functions of non-negative variables, the problem is called a linear programming problem, and the method of solution is the simplex method [56].

The general theory of mathematical programming was developed by Kuhn and Tucker [35]. The Kuhn-Tucker theory applies both the linear and nonlinear programming problems. The theory utilizes a generalization of the LaGrangian method of undetermined multipliers. It extends the LaGrangian method to inequality constraints. Kuhn-Tucker theory relies on concepts of convexity and duality.

In general, according to the theory, if a constrained objective function (primal program) is convex, then, for all $x > 0$, there exists an equivalent dual program, involving a dual function with a maximum at the primal minimum point. This theory of duality provides an indirect means of arriving at the optimum, since the dual program may be easier to solve than the primal and provides an easy way to check whether or not the optimum has indeed been reached.

Development of Geometric Programming

Many mathematical models involve variables raised to arbitrary powers. A great many theoretical laws and empirical results are expressed in such terms and others can be reduced to that form. It was the problem of minimizing the sum of component costs equal to design variables raised to arbitrary powers that led to the observation

by Zener [58] that such functions could be minimized almost by inspection, when the number of terms was one greater than the number of variables. These functions of the form shown in Equation 1.2 can be

$$y = \sum_{t=1}^{T} c_t \prod_{n=1}^{N} x_n^{a_{tn}} \tag{1.2}$$

classed as *generalized polynomials*. However, Zener and Duffin [60] established a theory in which the a_{tn} and x_n above could be any real number but the c_t were restricted to be positive. They called these functions *posynomials*. Following an approach akin to dimensional analysis, weights were chosen to make the product of component costs non-dimensional in the dual function. This product, Zener observed, was in fact the minimum cost.

Consultation with Duffin extended the theory to posynomials of an arbitrary number of terms [17], and related it [16] to Cauchy's inequality [24]. This theoretical connection to Cauchy's inequality, also called the *arithmetic-geometric mean* inequality, led to the name *geometric programming*.

Negative coefficients were forbidden the functions, and the sense of inequality was restricted, by the convexity requirement. Extension of geometric programming to negative coefficients and arbitrary inequalities demanded a release from convexity, accompanied necessarily by a release from the arithmetic-geometric mean inequality. The posynomial theory involved raising terms to fractional powers, thus the terms could not involve negative constants.

This extension was made, utilizing signum functions and Lagrangian theory, for the equality constraint case [55], and then for the inequality constraint case [47]. Unfortunately, this release from convexity also released the theory from simple convex duality. The primal and dual functions each may have several maxima, minima, or saddle points. It is this theory upon which the algorithm presented in this report is based. The present theory is also summarized in *Foundations of Optimization* [56].

CHAPTER II

GEOMETRIC PROGRAMMING

Posynomial Functions

Geometric programming was conceived as a means of optimizing the cost in engineering design problems in which the cost is a sum of terms expressing component costs. Zener [58] observed that such a sum may be easily minimized when each cost depends on products of the design variables raised to arbitrary but known powers. Cost functions of this type are of the form

$$y = \sum_{t=1}^{T} c_t \prod_{n=1}^{N} x_n^{a_{nt}} \tag{2.1}$$

in which, $c_t \geq 0$, $x_n \geq 0$, and a_{nt} may be any real number. If the non-negativity restriction is removed on c_t, Equation (2.1) represents a generalized polynomial. However, for reasons which will become clear below, Zener [19] restricted the $c_t \geq 0$ and named these particular functions *posynomials*.

The posynomial can be transformed by a change of variables

$$e^{z_n} = x_n \quad , n=1,\ldots,N$$

$$\text{when } x_n > 0 \tag{2.2}$$

such that Equation (2.1) becomes

$$y = \sum_{t=1}^{T} c_t e^{\sum_{n=1}^{N} a_{tn} z_n} \tag{2.3}$$

For posynomials, Equation 2.3 is convex and therefore has a global minimum. If the sign restriction on c_t is lifted, a stationary point can be found, but there is no guarantee that this point is the global minimum. Formal optimization methods, proofs, and examples are provided by Duffin, Peterson, and Zener [19] and Wilde and Beightler [56]. An introductory summary of basic geometric programming concepts using the notation of Beightler, Crisp, and Meier [2] is provided in the remainder of this chapter.

Optimizing Posynomials

To obtain the minimum of a convex function, the first partial derivatives of y are taken with respect to each of the variables $\underset{\sim}{x}$, and the resulting nonlinear equations are set equal to zero (signifying that the first partial derivatives vanish) as shown in Equation 2.4.

$$\frac{\partial y}{\partial x_k} = \frac{1}{x_k^*} \sum_{t=1}^{T} a_{tk} c_t \prod_{n=1}^{N} (x_n^*)^{a_{tn}} = 0 , \quad k=1,\ldots,N \tag{2.4}$$

Following the conventional procedure, weights shown as w_t in Equation 2.5 and representing the ratio of the individual terms in Equation 2.1 evaluated at the optimum $\underset{\sim}{x}^*$ to the optimum cost y^* are defined.

$$w_t = \frac{c_t \prod_{n=1}^{N} (x_n^*)^{a_{tn}}}{y^*} , \quad t=1,\ldots,T \tag{2.5}$$

By definition, the weights sum to unity

$$\sum_{t=1}^{T} w_t = 1 \tag{2.6}$$

Assuming that y^* and x_k^* are non zero, Equations 2.5 and 2.4 give

$$\sum_{t=1}^{T} a_{tk} w_t = 0 , \quad k=1,\ldots,N \tag{2.7}$$

Equations 2.6 and 2.7 are linear in w_t and do not depend on the cost coefficient c_t. Since the weights sum to unity

$$y^* = \prod_{t=1}^{T} (y^*)^{w_t} = \prod_{t=1}^{T} \frac{c_t \prod_{n=1}^{N} (x_n^*)^{a_{tn}w_t}}{w_t}$$

$$y^* = \prod_{t=1}^{T} \left(\frac{c_t}{w_t}\right)^{w_t} \prod_{t=1}^{T} \prod_{n=1}^{N} (x_n^*)^{a_{tn}w_t} \tag{2.8}$$

Inverting the order of the products in the double product term in Equation 2.8 and applying Equation 2.7, one obtains

$$\prod_{t=1}^{T}\prod_{n=1}^{N}(x_n^*)^{a_{tn}w_t} = \prod_{n=1}^{N}\prod_{t=1}^{T}(x_n^*)^{a_{tn}w_t}$$

$$= \prod_{n=1}^{N}(x_n^*)^{\sum_{t=1}^{T} a_{tn}w_t} \qquad (2.9)$$

$$= 1$$

Therefore Equation 2.8 can be simplified to become

$$y^* = \prod_{t=1}^{T}\left(\frac{c_t}{w_t^*}\right)^{w_t^*} \qquad (2.10)$$

To find y^*, the w_t values are calculated from Equations 2.6 and 2.7 and combined with the known c_t values. The optimum value of y can thus be found without knowing the optimum values of the decision variables. If the minimum total cost, y^*, is acceptable, Equation 2.5 can be used to find the optimum values of the decision variables. Thus, the total cost of several designs can be compared before the values of the individual decision variables are obtained.

When applied under favorable conditions, geometric programming

exhibits remarkable ability to locate optima. For illustrative purposes, consider the following unconstrained optimization problem. In this simple example problem, it is desired to find the dimensions of a cylindrical storage tank which would minimize the construction and operating costs for the tank and filling system. Let the total cost be denoted by C_T

$$C_T = c_1 \frac{k}{\frac{\pi d^2 h}{4}} + c_2 \frac{\pi d^2}{4} + c_3 \pi dh \qquad (2.11)$$

where d and h are the diameter and height of the tank and c_1, c_2, and c_3 are, respectively, the unit costs of filling the tank, material for the base, and material for the sides. The quantity k is the total volume to be supplied from storage in a period of time such as a year. In this crude example, the operating costs are represented by the first term while the construction cost is represented by the latter two terms.

Grouping the constants together, Equation 2.11 can be rewritten in the following form

$$C_T = \left(c_1 \frac{4k}{\pi}\right) d^{-2}h^{-1} + \left(c_2 \frac{\pi}{4}\right) d^2 + (c_3 \pi)\, dh$$

$$= c_1' d^{-2}h^{-1} + c_2' d^2 + c_3' dh \qquad (2.12)$$

Equation 2.12 is similar in form to Equation 2.1, and, as all $c_t' \geq 0$, geometric programming may be used to solve the problem. Applying Equations 2.6 and 2.7, one obtains

$$\begin{aligned} w_1 + w_2 + w_3 &= 1 \\ -2w_1 + 2w_2 + w_3 &= 0 \\ -w_1 + w_3 &= 0 \end{aligned}$$

Solving the equations above for the weights, the following results are obtained.

$$w_1 = 2/5, \quad w_2 = 1/5, \quad w_3 = 2/5$$

It is interesting to note that, without knowing the individual cost coefficients for filling the tank or constructing it, it is possible to learn that, at the optimum size, 40% of the total cost is incurred in filling the tank, 40% in constructing the sides, and 20% in constructing the base. This distribution of costs remains invariant regardless of the value of the c_t's. These *invariant properties* are characteristic of many geometric programming solutions and provide additional insight to the analyst.

To evaluate the minimum cost, C_T^*, the cost coefficients and the weights are substituted into Equation 2.10. If the coefficients are assumed to be c_1=10, c_2=6, and c_3=2 and k is assumed to be 80×10^4, the resulting C_T^* is found to be

$$C_T^* = \$5191$$

The values of the variables may be computed using any two weights of the type given in Equation 2.5. Using the term for w_2

$$(1.5)d^2 = (0.2)(5191)$$

$$d = 14.8$$

likewise using the third term

$$2dh = (0.4)(5191/\pi)$$

$$h = 22.3$$

Constrained Optima

The general geometric programming problem is to minimize a posynomial subject to constraints of the form

$$y_m \leq 1, \qquad m = 1,\ldots,M \tag{2.13}$$

where y_m is also a posynomial. The constrained optima problem is solved by extending Equation 2.10 to include each term of the constraints. A weight for each constraint is added to the weights for each term as outlined elsewhere [19,56].

The ability to handle constraints makes geometric programming applicable to a wide range of engineering problems. Some constraints are natural forms of the problem, whereas others result from the process of converting functions to posynomial form. For example, the function

$$y(x) = (x_1^{-1} + x_2)^a \ (x_1 + x_2^{-2})^b$$

cannot be minimized directly since it is not a posynomial. However, it can be replaced by the equivalent problem

$$\text{Minimize } y(x) = x_3^a x_4^b$$

$$\text{Subject to } x_1^{-1} x_3^{-1} + x_2 x_3^{-1} \leq 1$$

$$x_1 x_4^{-1} + x_2^{-2} x_4^{-1} \leq 1$$

This new problem falls into the framework of geometric programming.

Degrees of Difficulty

Equation 2.6 and 2.7 provide N+1 linear dual variables, where N is the number of primal variables. There are, however, T dual variables in the dual equations, where T is the number of primal terms. The dual function is then unique if

$$T = N+1 \tag{2.14}$$

If

$$T < N+1 \tag{2.15}$$

the primal problem is generally unconstrained and therefore not considered by geometric programming.

The degree of difficulty, D, is defined as

$$D = T-(N+1) \tag{2.16}$$

As D increases, the number of independent dual variables (equal to D) also increases. The dual problem is then a function of D independent and N+1 dependent variables.

If an optimum point exists, the minimum of the primal problem equals the maximum of the dual. The task of finding this point is a numerical problem increasing in difficulty as D gets larger, thus explaining the meaning of the term "degree of difficulty".

$$C_T = (c_1 4K) / (\pi d^2 h) + (c_2 \pi d^2) / 4 + c_3 \pi d h + c_4 h \qquad (2.17)$$

with $c_4 = 1$. If, as in the previous example, this objective describes the cost of building and operating a water storage tank, the additional term could be considered the cost of adding supports to the sides. Using geometric programming, the weights are introduced following Equations 2.6 and 2.7.

$$\begin{aligned} w_1 + w_2 + w_3 + w_4 &= 1 \\ -2w_1 + 2w_2 + w_3 &= 0 \\ -w_1 + w_3 + w_4 &= 0 \end{aligned}$$

The minimum cost C_T^* cannot be computed until the optimum weights are known. Using the three equations above, it is possible to express each of the weights in terms of any one of the weights. Expressing all in terms of w_3, the following are obtained.

$$w_1 = \frac{1}{3}(1 + \frac{1}{2} w_3);\ w_2 = \frac{1}{3}(1 - w_3);\ w_3 = w_3;\ w_4 = \frac{1}{3}(1 - \frac{5}{2} w_2)$$

Any choice of w_3 will produce values for the other weights that will satisfy Equations 2.6 and 2.7. The problem is now to select the optimum weights from among those possible using Equation 11.

Equation 10 holds for any number of degrees of difficulty, and the minimum cost will be given as a function of w_3 alone. This equation will be denoted as $d(w_3)$, the *substituted dual function.*

$$d(w_3) = \left(\frac{4Kc_1}{\pi(1/3(1+1/2w_3)}\right)^{\left(\frac{1+\frac{1}{2}w_3}{3}\right)} \left(\frac{\pi c_2}{4[1/3(1-w_3)]}\right)^{\left(\frac{1-w_3}{3}\right)}$$

$$\left(\frac{c_3\pi}{w_3}\right)^{w_3} \left(\frac{c_4}{[1/3(1-5/2w_3)]}\right)^{\left(\frac{1-5/2w_3}{3}\right)}$$

Duffin [19] has proved that the substituted dual function is maximized by the optimum weight w_3^* and that this maximum value is equal to the minimum cost C_T^*.

$$C_T^* = d(w_3^*) = \max_{w_3} [d(w_3)]$$

In general, Duffin showed that a sufficient condition for C_T to be minimized is that the dual function defined by

$$d(w_1, \ldots, w_T) = \prod_{t=1}^{T} \left(\frac{c_t}{w_t}\right)^{w_t}$$

be maximum with respect to w_t, subject to the conditions given in Equations 2.6 and 2.7.

The numerical value of w_3^* can be found by obtaining the first derivative of the logarithm of the substituted dual function and setting this derivative equal to 0. When this is done, the optimum value of w_3 is found to be

$$w_3^* = 0.395$$

The remaining optimum weights can be found using the relations shown in Equation 11.

$$w_1 = 0.399; \quad w_2 = 0.202; \quad w_3 = 0.395; \quad w_4 = 0.004$$

The weights can then be used as before to find the optimum values of d, h, and C_T^*.

$$d = 14.94 \quad h = 21.94 \quad C_T^* = 5213$$

Computer Algorithm

Edwards [20] has developed a computer program, which he calls the Geometric Programming System (GPS), to solve generalized geometric

programming problems. GPS with its eight subprograms is written in Fortran IV for the IBM 360/65 at Texas A&M University. If degrees of difficulty exist in the problem to be solved, the program calls a "pattern search" algorithm developed by Letto [37] to maximize the dual function. GPS provides the following output:

1. The primal problem.
2. The dual matrix.
3. The value of the dual function and dual variables.
4. The solution to the primal problem including the optimum value of primal variables and the value of each term and each row.
5. The absolute and relative difference between the primal and dual functions.
6. The elapsed time used to solve the problem.

The next chapter describes the computer program in more detail.

Generalized Polynomials

Passy and Wilde [47] extended the original posynomial form to permit negative coefficients and reversed inequalities (constants ≥ 1). This extension, however, releases the problem from its convexity form permitting the location of stationary points but no longer guaranteeing a global minimum.

Consider as the generalized polynomial

$$y_m(x) = \sum_{t=1}^{T_m} \sigma_{mt} c_{mt} \prod_{n=1}^{N} x_n^{a_{mtn}} \qquad (2.18)$$

$$m = 1, \ldots, M.$$

where $\sigma_{mt} = \pm 1$ to generalize the sign of c_{mt}, and all $c_{mt} > 0$. T_m is the number of terms in y_m.

The primal problem is then to

Minimize $y \equiv y_1^*$

subject to the constraints

$$y_m \leq \sigma_m \ (= \pm 1), \ m = 2,\ldots,M.$$

and

$$x_n > 0, \ n = 1,\ldots,N.$$

The dual problem contains T variables ω_{mt} satisfying a revised normality condition

$$\sum_{t=1}^{T_1} \sigma_{1t}\omega_{1t} = \sigma_1 (\equiv \pm 1), \tag{2.19}$$

N revised orthogonality conditions:

$$\sum_{m=1}^{M} \sum_{t=1}^{T_m} \sigma_{mt} a_{mtn} \omega_{mt} = 0, \quad n = 1,\ldots,N, \tag{2.20}$$

T nonnegativity conditions

$$\omega_{mt} \geq 0, \quad m = 1,\ldots,M$$

$$t = 1,\ldots,T_m$$

and M - 1 inequality constraints

$$\omega_{mo} \equiv \sigma_m \sum_{t=1}^{T_m} \sigma_{mt}\omega_{mt} \geq 0 , \qquad (2.21)$$

$$m = 2,\ldots,M$$

where $$T = \sum_{m=1}^{M} T_m ,$$

and σ_1 can be shown to be of the same sign as y^*. The reader will note that the weight are designated ω_{mt} in the generalized case to differentiate it from the posynomial case.

Then the dual function is constructed as follows:

$$d(\underset{\sim}{\omega}) \equiv \sigma_1 \left[\prod_{m=1}^{M} \prod_{t=1}^{T_m} \left(\frac{c_{mt}\omega_{mo}}{\omega_{mt}} \right)^{\sigma_{mt}\omega_{mt}} \right]^{\sigma_1} , \qquad (2.22)$$

where

$$\omega_{10} \equiv +1,$$

and, by L'Hôpital's rule,

$$\lim_{\omega_{mt} \to 0} \left(\frac{c_{mt}\omega_{mo}}{\omega_{mt}} \right)^{\sigma_{mt}\omega_{mt}} = 1.$$

If all $\sigma > 0$, the primal problem is convex in its e-transformation, and at the optimum

$$y_1(\underset{\sim}{x}) \geq y_1(\underset{\sim}{x}^*) = d(\underset{\sim}{\omega}^*) \geq d(\underset{\sim}{\omega}). \tag{2.23}$$

In a nonconvex case, the local minima of the primal function correspond to the dual critical points such that

$$y_1(\underset{\sim}{x}^0) \equiv \min y_1(\underset{\sim}{x}) = \max d(\underset{\sim}{\omega}) \equiv d(\underset{\sim}{\omega}^0). \tag{2.24}$$

These points may be local maxima, minima, or saddle points.

After the proper values for the dual function are found, the primal variables can be solved for by the following equations, which are linear in ln x and are analogous to Equation 2.5 for the convex case.

$$c_{1t} \prod_{n=1}^{N} x_n^{a_{1tn}} = \omega_{01}\sigma_1 y_1^{0}, \quad t = 1,\ldots,T_1 \tag{2.25}$$

$$c_{mt} \prod_{n=1}^{N} x_n^{a_{mtn}} = \frac{\omega_{mt}}{\omega_{mo}}, \quad t = 1,\ldots,T_m \tag{2.26}$$

$$m = 2,\ldots,M.$$

It can be seen from Equation 2.25 that σ_1 has the same sign as y_1^0. There are $T_M > N$ equations, so N can always be found, and the value of each primal variable can be determined.

CHAPTER III

COMPUTER ALGORITHM

Generalized Notation

As a part of this research, a computer algorithm for solving geometric programming problems was developed. This program is discussed in this chapter. The notation employed is the x - ω notation for the generalized problem [2]. Since there are no lower case characters on the line printer, capital letters in the text notation are prefaced by CAP- in the computer code. (See Appendix A). Thus, M represents m, and CAPM means M. However TSUBM(M) stands for T_m.

Furthermore, computer arrays begin with subscript 1, whereas the texts refer to, for instance, ω_{01}. The notation of this report, however, corresponds to the notation of the algorithm, with but one exception. In that case, the array ω_{mt} becomes a vector w_i, where

$$\omega_{mt} = w_i \qquad t = 1,\ldots,T_m \qquad m = 1,\ldots,M, \qquad (3.1)$$

$$\{1 = t + \sum_{i=0}^{m-1} T_i\}$$

if $T_0 = 0$, and also

$$\omega_{mo} = w_{T_M + m + 1}, \qquad m = 0,\ldots,M.$$

Main Program

GPS, the main program, and its eight subprograms have been written in Fortran IV for the IBM 360/65 at Texas A&M University. See Appendix D for a description of subroutine interfaces.

GPS (for Geometric Programming System) reads in data and prints out the primal problem. (See Figure 3.1). The augmented matrix of coefficients of the dual variables is constructed in array B, and subroutine MAT is called to solve the system, if possible, or to place an identity matrix in the left hand side if it is not. From Equations 2.19 and 2.20, it will be seen that the system of dual equations occupies a matrix of $N + 1$ rows and T_M columns. Thus if degrees of difficulty exist, that is, $D \equiv T_M - (N + 1) > 0$, there will be D columns on the right side of the matrix which represent independent dual variables. The $T_M + 1$st column is the constant vector in the augmented matrix which is printed.

If degrees of difficulty exist, a pattern search algorithm by A. R. Letto [37] is called to maximize the dual function as a function of the D independent dual variables, constrained by the natural nonnegativity requirement on the $T_M + M$ dual variables. Subroutine VAL is then called to evaluate the dual function at this maximum point, and the dual solution is printed.

From Equations 2.25 and 2.26, it has been shown that the equations involving the primal variables are linear in $\ln x$. Matrix B is loaded with the linearized equations, and once again MAT is called to solve the system. This time the matrix has T_M rows and N

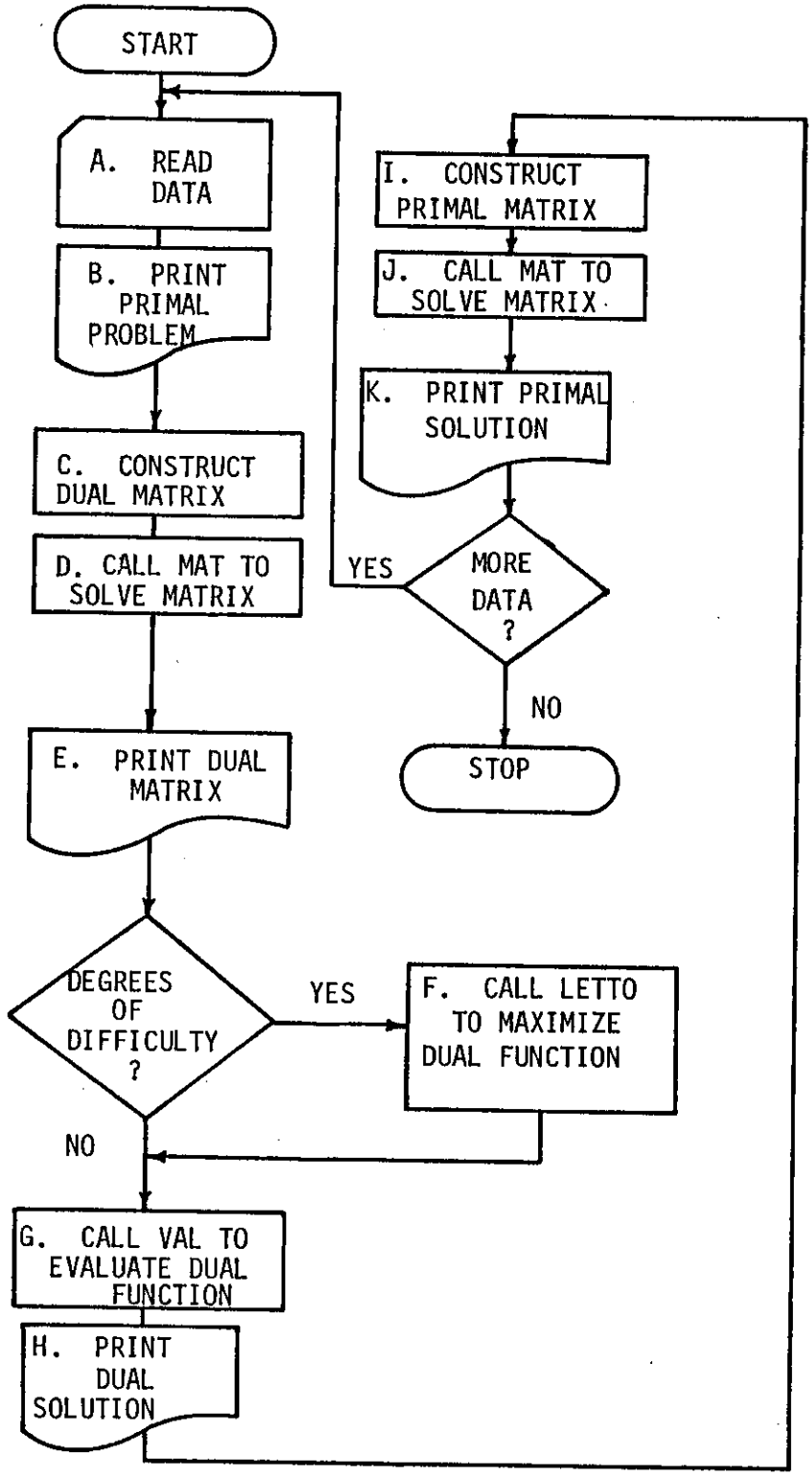

Figure 3.1 Program GPS.

columns, $T_M > N$. However, it is now known whether or not all the primal variables appear in the top N rows; so the routine solves for them wherever they appear. The primal solution is then printed, and, if another problem does not follow, execution is terminated.

Subroutine MAT

MAT is called twice, once to solve the dual matrix, then to solve the primal matrix. The general characteristics of the routine are noted above. The method of solution is simply algebraic addition of equations, after the method of Gauss-Jordan.

The pivot element (see Figure 3.2) referred to is simply the element on the upper left to lower right diagonal. The subtraction of rows is performed in the usual weighted manner, so that the elements in the pivot element's column are summed to zero.

The algorithm above (10) forms B into an upper-triangular matrix, and below (10) it finishes the job of creating the left side identity matrix. Note that the exchange of rows and columns, which always occurs downward and to the right, effectively insures that no information concerning the solution shall be lost if it is sought.

Subroutine LETTO

When the dual matrix has been constructed, if degrees of difficulty exist, LETTO is called. The principal elements given to LETTO are the dual matrix B and the number of degrees of difficulty. The D independent dual variables become the pattern search variables, and all the dual variables are invoked as constraints (see Output). LETTO calls VAL, SUMS, SUMC, and OUTPUT. It manipulates the D independent

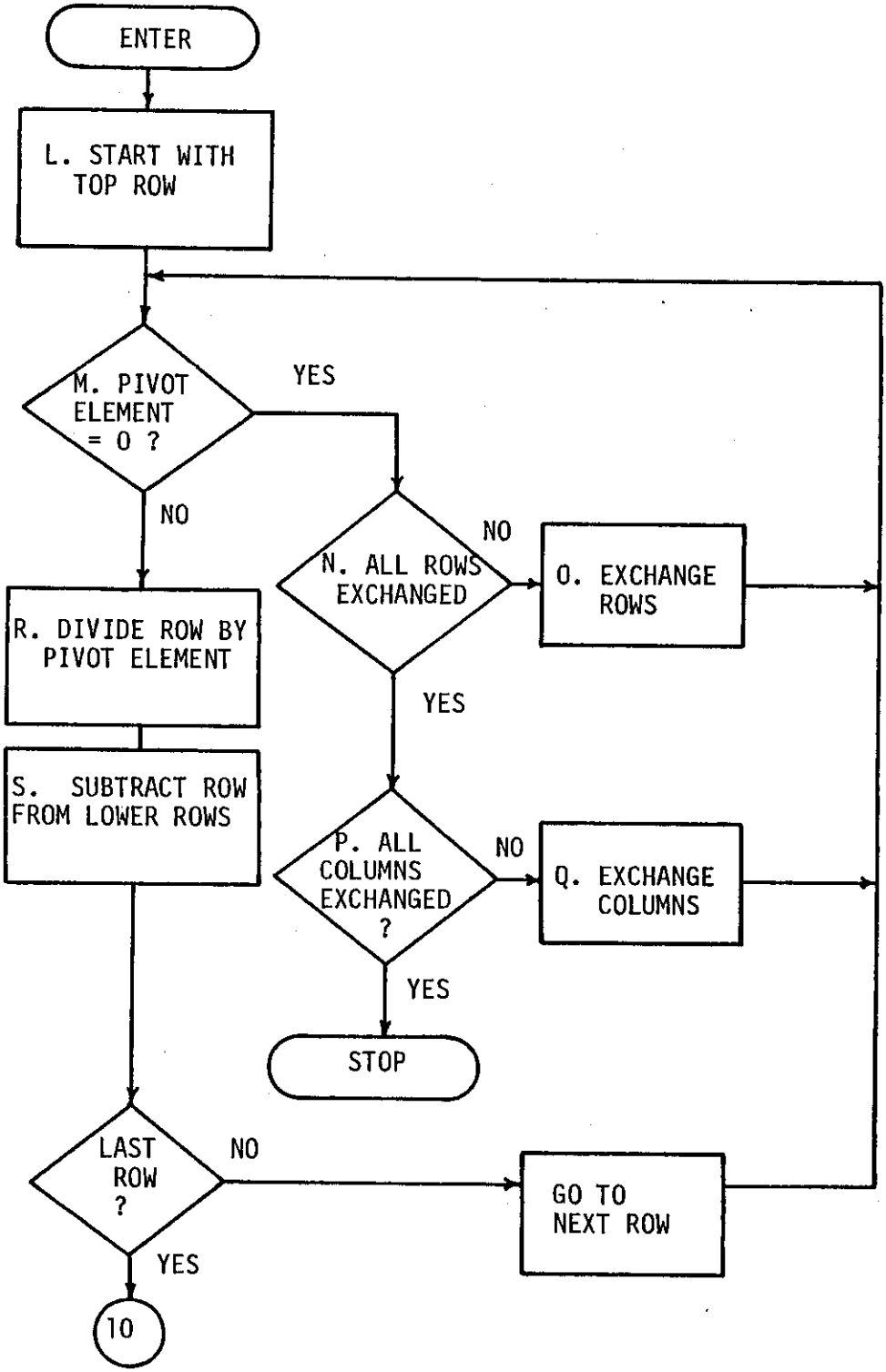

Figure 3.2 Subroutine MAT Flow Diagram

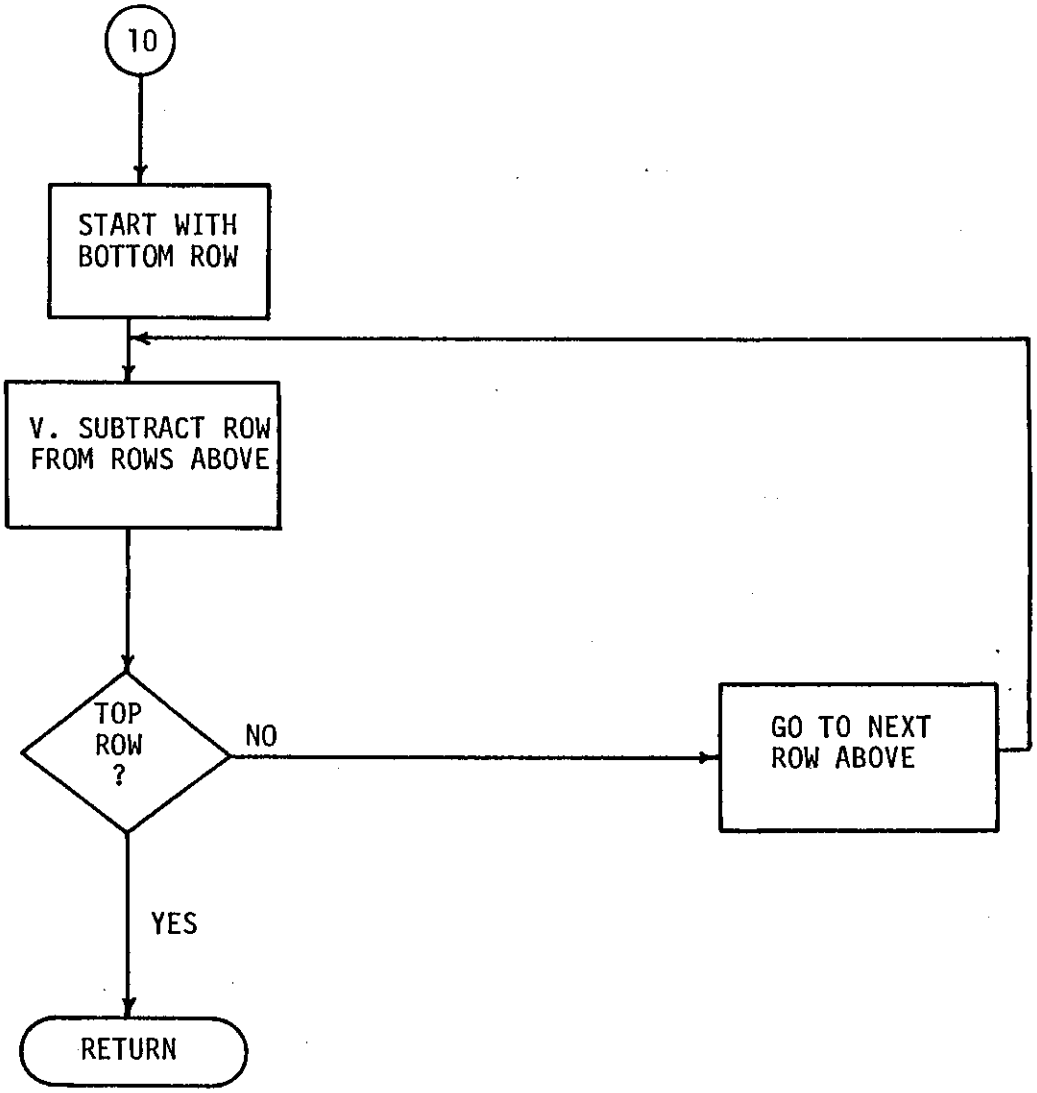

Figure 3.2. Continued.

dual variables and, evaluating the dual function and constraints through VAL, proceeds to search out a maximum of the dual.

The search method is after R. Hooke and T. A. Jeeves [25] and basically involves two kinds of moves. The first is exploratory search, in which each variable is perturbed slightly from its starting point. If no improvement is noted in the dual function, the variables are perturbed with a smaller step size. This may be repeated until the step size falls below some lower limit, when the search is successfully terminated. In the printout, this is called "local search." If the perturbation of any variable produces an improvement (larger value) over the base point, the remaining variables are perturbed around this new base point. The difference between the final and original base points defines a vector which lies "uphill" and approximates the route of steepest ascent (direction of largest gradient).

The second kind of move, the pattern move, proceeds in the direction of this vector. Each step goes twice the distance of the preceding step. This multiplication factor gives the method its power. When the last point found is worse than the previous point, the method reverts to exploratory search about the previous point. Thus the method never proceeds to a worse point.

When a constraint is contacted (some $\omega < 0$), a gradient summation technique is employed to enable the routine to proceed uphill along the edge of the constraint. If the starting point is infeasible, the sum of all the constraints becomes the objective

function to be maximized. Feasibility is reached when all constraints are positive. The Letto algorithm performs rapidly and seems always to find an optimum if one exists, and a global optimum if the region is concave.

Subroutine VAL

VAL receives the values of the independent dual variables, if there exist degrees of difficulty, and solves for all the dual variables from the matrix B, which has been simplified so that the calculation can be performed rapidly. This subroutine then calculates the value of the dual function, knowing $\underset{\sim}{\sigma}$, $\underset{\sim}{\omega}$, and all c_{mt}.

Other Subroutines

Subroutine BLOWUP is called whenever an obvious error in data is read. It explains the error and terminates execution, since the next problem could not be read in while data for the present problem is still in line.

Subroutine OUTPUT prints out the values of the dual function, independent dual variables, and all constraints at each step in the pattern search. The constraints are in fact all the $\underset{\sim}{\omega}$ in order. (See Equation 3.13). $\omega_{10} = W_{T_{M+1}} \equiv 1$ is included for consistency.

The BLOCK DATA subroutine initializes certain variables. Note that $\underset{\sim}{X}$ is initialized to 0.2 for the first problem, but is never reinitialized, except for $\underset{\sim}{X}_1 = 0$ for any zero degree of difficulty problem. This is done so that similar problems can be read in following one another, and the time required to find the close optimum in the pattern search will be small. Indeed, except in such cases,

any value for $\underset{\sim}{X}$, $0 < \underset{\sim}{X} < 1$, is about as good as any other, since it would be very difficult to pre-determine the dual optima.

Input

The program will handle generalized polynomials for which $M \leq 10$; $N \leq 10$; $T_m \leq 10$, $m = 1,\ldots,M$; $D \leq 20$.

The input consists of:

Card 1; (22I2 format)

M, N, $(T_m, m = 1,\ldots,M)$, $(\sigma_m, m = 1,\ldots,M)$

Card 2; (A80 format)

Title of problem

Card 3; (E 10.0, I1 format)

XLLIM, the lower limit on variable moves in the Letto pattern search routine, and

LPRINT. If LPRINT = 0, the pattern search values are printed out at each step, with all messages.

If LPRINT = 1, these values are printed out only at beginning and end of the Letto routine. All messages are printed out at each step.

If LPRINT = 2, only feasibility messages are printed out.

If LPRINT = 3, only final values are printed out.

The next cards read in the coefficients and exponents of each term of the primal function and all constraints in order. For example, the first two terms of the primal function might be:

$$c_{11} x_1^{a_{111}} x_2^{a_{112}} x_3^{a_{113}} + c_{12} x_3^{a_{123}} x_5^{a_{125}} x_2^{a_{122}}$$

Then the cards would be as follows:

Card 4 (E 25.0 format) (If E is punched, it must be right justified.)

c_{11}

Cards 5-7 (I2, E 16.8, I2 format)

1, a_{111}, 0

2, a_{112}, 0

3, a_{113}, 1 (1 denotes last variable in term)

Card 8 (E 25.0 format)

c_{12}

Cards 9-11 (I2, E 16.8, I2 format)

3, a_{123}, 0

5, a_{125}, 0

2, a_{122}, 1

and so forth to the last term in the last row of the generalized polynomial.

If there is another problem to be run at this time, succeeding

cards should follow the same order as before, starting with Card 1.

After the last problem, one final card should contain 1 in column 2 and otherwise be blank.

Output

A typical output is given in Appendix B. The line below the main heading is the title of the problem, read in on the second card. The generalized polynomial is then printed out, with the horizontal lines separating the objective function and constraints. The symbol .L E.on the right implies that the constraint $\overset{<}{-} \sigma$ ($\equiv \overset{+}{-} 1$).

On a new page the augmented dual matrix is printed out after having the identity matrix placed in its left side. The rightmost column is the constant or result column. The next D columns to the left represent the independent dual variables.

The pattern search output is begun if D> 0. The print option determines how much is printed out.

The value of the dual function and of all the dual variables is printed out next. Then the primal solution, including the optimal values of the variables and the values of each term and each row of the generalized polynomial primal problem, evaluated at those values, is printed out.

The absolute and relative difference between the primal and dual functions is shown. If the problem is convex, the optimal value can be shown to lie between the two functional values; *i.e.*, an upper and lower limit and maximum error is determined. The elapsed time finishes the print-out.

Error Messages

The error messages from subroutine BLOWUP are printed out after the data in question is printed, and should be self-explanatory.

One message from MAT can be printed to signify that the matrix in question is singular. That is, there are zeros in all of the elements to the right and below a certain pivot element, and thus the matrix cannot be solved. If the message occurs after the primal problem is printed out, the singularity was found in the dual matrix. If it occurs later, after the dual variables are printed out, the fault lies in the primal matrix. Job execution stops, but the next problem may be read in.

The message SIG(1) = -SIG(1) signifies that the program has realized that the sign guessed for the objective function at a minimum was incorrect. The value is changed and execution continues without prejudice to the results.

.2)

CHAPTER IV

WATER FILTRATION THEORY

The Filtering Process

The purpose of this chapter is to provide a basis for the model development associated with the comprehensive example optimization problem involving water filtration which is described in Chapter V. The solution of this problem is discussed in Chapters V and VI.

Water filtration is the process of passing water through a porous media to remove turbidity, suspended matter, and some colloidal material. The turbidity and suspended matter may be present in the raw water or may result from some precipitation action preceding the filtration process in the water treatment sequence. Particle removal in filtration involves two mechanisms -- a physical transport mechanism and a chemical attachment mechanism. Most filtration theory and all of the mathematical models are based on the physical mechanism [4,11,31,32,29]. The chemical mechanism has been suggested [44], but it has not been mathematically modeled. A review of some of the important literature follows.

Physical Removal Mechanisms

Iwasaki, [33] the founder of modern filtration theory, developed three fundamental equations describing the filtration process

$$-\frac{\partial C}{\partial L} = \lambda C \tag{4.1}$$

$$\lambda = \lambda_0 + c\sigma \tag{4.2}$$

$$\frac{\partial C}{\partial L} + \frac{\partial \sigma}{\partial L} = 0 \qquad (4.3)$$

in which, C = concentration of suspended particles, L = filter depth, λ = coefficient of proportionality (also termed the filter coefficient), σ = volume of suspended matter retained per unit volume of filter, and c = a constant.

Equation 4.1 states that the amount of matter arrested in a lamina is proportional to the amount that enters the lamina. Equation 4.2 states that the change in the proportionality factor or filter coefficient is directly proportional to the density of the matter filling the voids. Equation 4.3 is a mass balance or continuity equation stating that the decrease of flow through a lamina is equal to the increase of the density therein.

Ives [31] modified Equation 4.2 stating that the filter coefficient, λ, first increases linearly with the specific deposit, σ, and then decreases nonlinearly.

$$\lambda = \lambda_o + k\sigma - \frac{\phi\sigma^2}{p_o - \sigma} \qquad (4.4)$$

Ives also modified Equation 4.3 to

$$-\frac{\partial C}{\partial L} = \frac{1-f_\sigma}{v} \frac{\partial \sigma}{\partial t} \qquad (4.5)$$

and added a new equation to describe the headloss.

$$\frac{\partial h}{\partial L} = \left(\frac{dh}{dL}\right)_o + k\sigma \qquad (4.6)$$

where λ_o = initial filter coefficient, p_o = porosity of clean bed, ϕ = a filter parameter, t = filtration time, f_σ = self porosity of deposited solids, h = headloss, k = headloss constant, v = approach velocity, and $(dh/dL)_o$ = initial (clean bed) headloss per unit depth.

Others including Sholji [51] and Deb [11] have suggested modifications to the work of Iwasaki and Ives, but the basic principles have not been changed. Deb [10] used a computerized finite-difference method to obtain a numerical solution to his resulting nonlinear partial differential equations. Deb's model is, however, restricted to homogeneous, unisize, granular media which somewhat limits its practical value.

Camp [4] took another approach to developing a mathematical model combining a modified form of the Kozeny equation of laminar flow [34] with Stein's refinement [52] of the Iwasaki model. Camp suggested two hydraulic gradient equations.

$$i = \frac{k(1-p_o+\sigma)^2}{(p_o-\sigma)^3(d+\Delta d)^2} \tag{4.7}$$

$$i = \frac{k(1-p_o)^2}{d_o^2 p_o^2 (p_o-\sigma)} \tag{4.8}$$

where i = hydraulic gradient (dh/dL), d_o = grain size at the start of filtration, Δd = increase in grain size at a particular time during a filter run in which the value of k is known, $k = (j\ u/g)v$, j being an overall characteristic of the filter bed media, u is the

kinematic viscosity of the fluid, g is the gravitational constant, and v is the approach velocity.

Equation 4.7 applies where the clogging or removal mechanism is due to the formation of a sheath on each grain. Equation 4.8 describes the situation where the clogging phenomena is due primarily to straining.

Laboratory filtration experiments [45,30] designed to compare the various equations of Ives, Camp, and others have shown a wide variance in effluent quality predictions. Such differences are probably due in part to the failure of the models to include the chemical mechanisms.

Chemical Removal Mechanisms

O'Melia and Struum [44] have written a comprehensive review of chemical filtration mechanisms. These mechanisms are usually related to Van der Waals forces and electrokinetic effects, but they have not been developed to a state where they can be expressed mathematically.

Van der Waals forces are the most powerful of the known attraction forces at the molecular level. These forces vary inversely at the seventh power of their difference of separation [9]. Electrokinetic effects are the result of an electric charge on the surface of colloidal particles caused by ionization or adsorption. Ions are centered around the solid particle in alternately charged layers. Electrokinetic repulsion between the outer layers tends to keep the particles dispersed. The presence of a charge on the surface of the filter media may, therefore, largely determine the stability of the colloidal dispersion.

In an attempt to include these chemical mechanisms with the physical, some researchers have posed a combination of theory and empirical results to describe the filtering process.

Empirical Studies

One of the more recent and comprehensive combinations of theory and empirical results was prepared initially by Hsiung for single media filters [26,27] and later extended to multimedia filters [8]. Hsiung proposed a new filter performance prediction theory based on a "random walk" analogy. If P represents the probability of penetration by a particle, the probability of a particle moving down to depth L at time t will be

$$P(L:rt,p) = \binom{rt}{L} p^L q^{(rt-L)} \tag{4.9}$$

where r = average rate of particle movement, p = probability of a success (particle moves one unit step downward), and q = probability of failure (the particle does not move).

The total number of particles arriving at L is

$$\partial m \sim P(L:rt,p)Q\, C_o \partial t \tag{4.10}$$

Then by Iwasaki's continuity equation, Equation 4.3, modified to let $\partial L = Q\, \partial t$

$$\frac{\partial C}{\partial L}\, Q\, \partial t \sim P(L:rt,p)Q\, C_o\, \partial t \tag{4.11}$$

or

$$\frac{1}{C_o} \frac{\partial C}{\partial L} \sim P(L:rt,p) \tag{4.12}$$

Thus particles are removed per unit depth of filter according to some probability function. Hsiung used a chi-square distribution to map this probability and proposed a deposit index, U, to translate filter performance data into a practical design parameter.

Hsiung then combined his theoretical aspects of filter performance with experimental data obtained from pilot plant studies for various grain sizes, flow rates, and influent concentrations. In the pilot plant studies, he first evaluated the effect of each variable on the filter performance for a given system holding the other variables constant. He then expressed this relationship in two performance curves [26,27]. The first curve compares U/L with $Q^a d^b t/L^c$, and the second compares $d^i(H_t-H_o)/Q^j C_o^k L^n$ with $Q^a d^b t/L^c$ in which, L = depth of bed, Q = filtration rate per unit area of filter, d = mean grain size of filter media, t = filtration time, H_t = headloss at time t, H_o = headloss at beginning of filtration, C_o = concentration of particles in influent, and a,b,c,i,j,k, and n are the parameters determined from the pilot plant studies. Plotted on log-log paper, both curves are concave. These performance curves can be developed for single or milti media filters. For multimedia, the equivalent grain size, d_e, is approximated by

$$d_e = x_1 d_1 + x_2 d_2 + x_3 d_3 \tag{4.13}$$

in which, d_1, d_2, and d_3 are the mean grain sizes of each media (assuming here that there are three sizes), and x_1, x_2, and x_3 are the percentage by volume of each media. A representative set of performance curves is shown in Figures 4.1 and 4.2. These curves were developed by Conley and Hsiung [8] to show a multimedia filter's typical pattern of performance for a synthetic muddy water after coagulation and settling. A secondary flocculant (polyacrylamide) was also applied immediately ahead of the filter. The addition of such polymers to the filter influent greatly increased filtration efficiency, but it also increased the head loss with resulting new performance curves. Thus, a filtration cost model developed around the performance curves cannot include the polymer as a variable since new constraints (based on the curves) would have to be used for varying amounts of polymer.

Multimedia Filters

Initially, filters were of the slow sand type with a filtration rate of 0.04 to 0.12 gallons per minute per square foot (gpm/sq ft). In the late 19th century, rapid sand filters were introduced operating at a rate of approximately 2 gpm/sq ft. Neptune-Microfloc has recently introduced a miltimedia filter of anthracite, sand, and garnet in layers of decreasing size. Slow and rapid sand filters have their finer grains at the top and perform most of the filtration in the top few inches. The miltimedia filter has its coarser grains at the top permitting filtration action through the entire depth of the filter (See Figure 4.3).

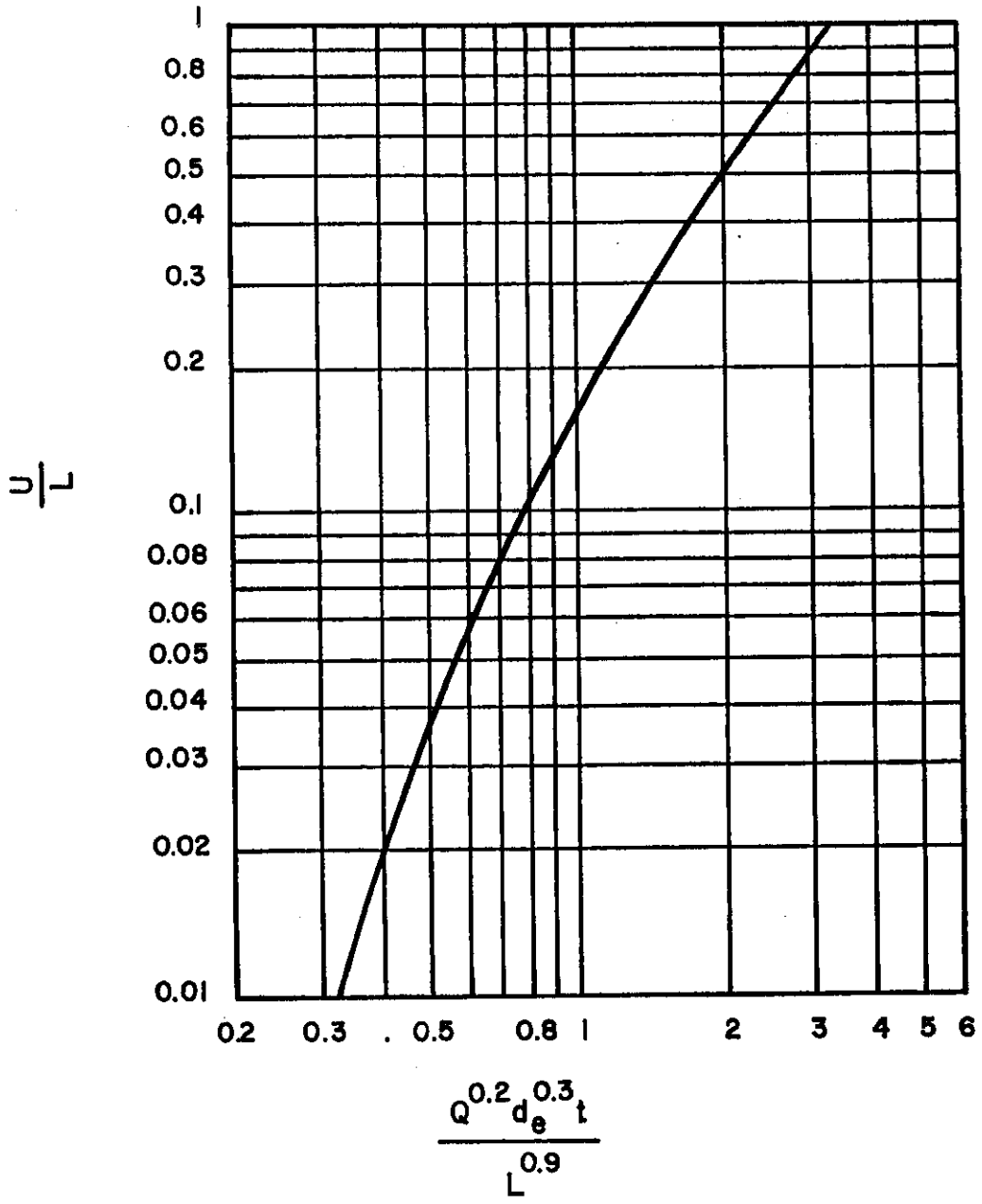

Figure 4.1 Performance Curve I for a Synthetic Muddy Water

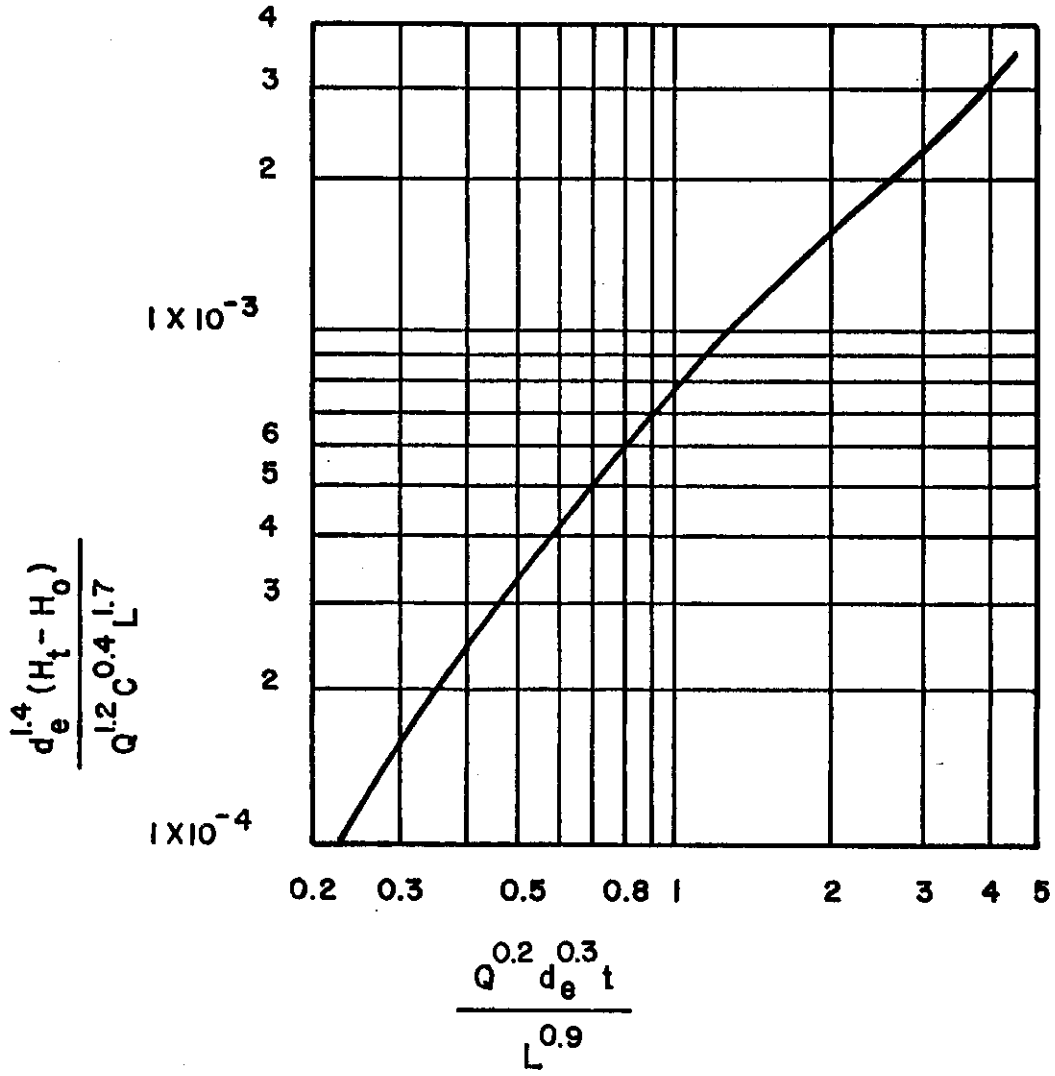

Figure 4.2 Performance Curve 2 for a Synthetic Muddy Water

ר ft.

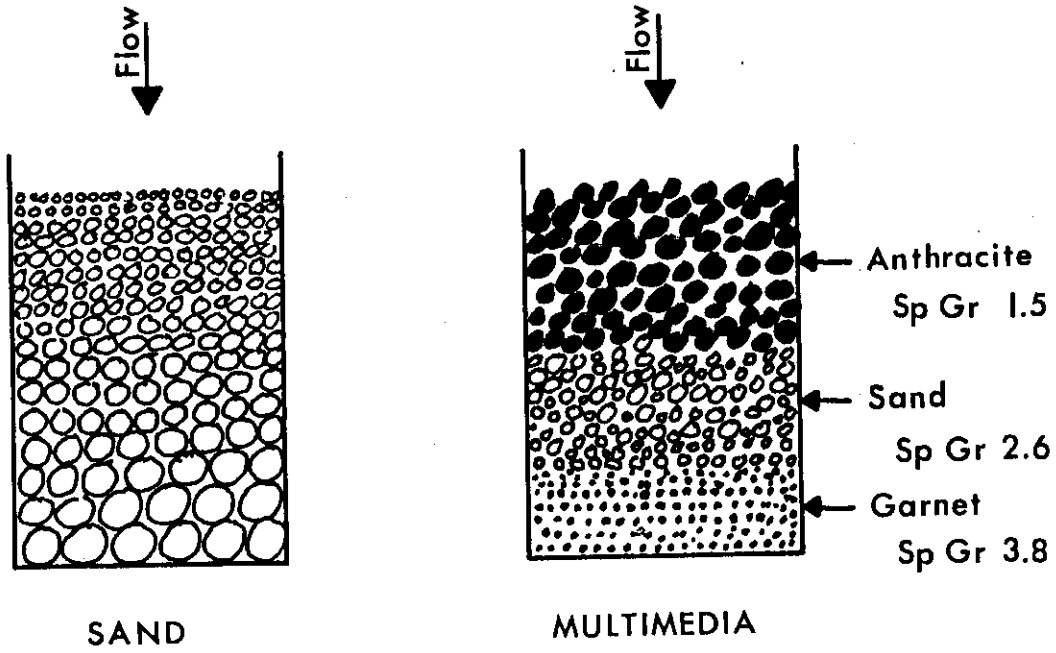

Figure 4.3 Comparison of Filter Media

Multimedia filters can operate at filtration rates up to 8 gpm/sq ft. These filters have gained wide acceptance [6,13,48,42] and offer an economical solution for treatment plants needing greater capacity than that of existing rapid sand filters [36,50]. In designing a multimedia filter, a wide range of choices are available for the specific gravity, size, and depth of each media layer. Selection of an optimum combination of media is dependent on several factors including turbidity loading, filtration rate, desired quality of effluent, and initial cost. In Chapter V, a cost model is developed to assist in the selection of an optimal media combination. First, however, the next chapter will review geometric programming, an optimization technique which will be used in finding a solution to the cost model.

CHAPTER V

FILTER COST MODEL

Reducing the Problem

There are a number of water filtration design problems to which a cost minimization technique could be applied. These problems range in scope from the design of a complete new facility to minor operating changes in an existing plant. A representative problem has been selected between these two extremes which adequately demonstrates the modeling technique and yet keeps the problem small enough to be tractable.

The problem chosen for presentation herein is the selection of an optimum trimedia to replace the rapid sand media in an existing filtration plant. A typical multimedia filter is shown in Figure 5.1. Such conversions have proven to be a very economical means of increasing plant capacity at minimal cost. For example, the water plant at Winnetka, Illinois, was converted from 6.0 to 12.0 million gallons per day (mgd) at less than 25 per cent of the cost of adding either new rapid sand or diatomaceous-earth filters [50]. Greenville, Texas, used multimedia filters to increase their plant capacity from 6.0 to 10.5 mgd at a cost of $200,000 as compared to $475,000 for new rapid sand filters [36].

In designing a multimedia filter, a wide range of choices are available for the size and depth of each media layer. Selection of an

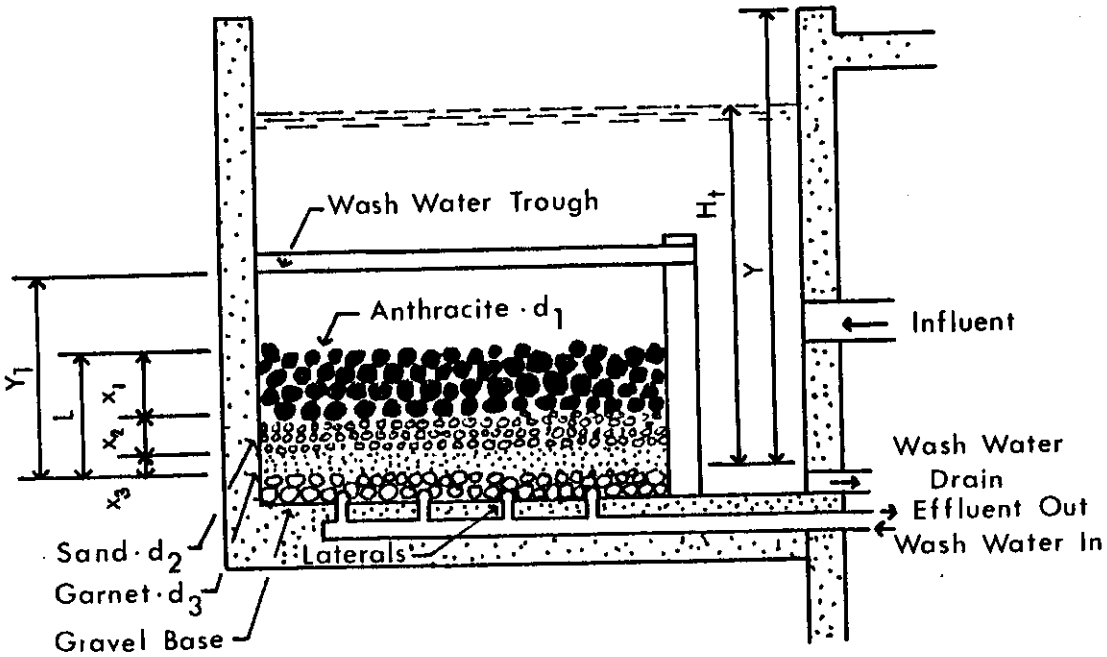

Figure 5.1 Multimedia Filter

optimum combination of media is dependent on many factors. These factors can be expressed in mathematical equations to form the objective function and constraints of the filter cost model.

Objective Function

The principal decision variables in the mathematical model are the depth of each media, flow rate, length of filter run, headloss, and the equivalent grain diameter. The diameter of each media could also be established as a decision variable with upper and lower bounds, but this greatly increases the number of variables requiring a large optimization program. To retain generality, the model will be presented in this chapter in its general form with all known variables including media diameter. Then in Chapter VI, some variables will be established as constants so that the Geometric Programming System (GPS) can be used to solve a sample problem.

The objective function in the mathematical model is designed to measure the effectiveness (in this case the cost) of media selection. The primary costs are the original cost of the media amortized over a specified period and the cost of the backwash water. Personnel operating costs, etc. are generally constant for all media and are therefore not included in the objective function. The objective function is expressed in dollars per million gallons based on a 30-day month average.

A series of payments which will repay the initial investment in

n years together with an annual interest of rate i on the uncovered portion can be found by [12]

$$\frac{i(1 + i)^n}{(1 + i)^n - 1} \tag{5.1}$$

Converting Equation 5.1 to a monthly series of payments and defining the resultant as R

$$R \equiv \frac{\frac{i}{12}(1 + i/12)^{12n}}{(1 + i/12)^{12n} - 1} \tag{5.2}$$

The objective function is then

$$\frac{[\frac{A R}{12}(x_1P_1 + x_2P_2 + x_3P_3) + \frac{43{,}200\ W\ A}{15t}]\ 10^6}{43{,}200\ Q\ A} \tag{5.3}$$

in which, A = surface area of filter (sq ft); x_1,x_2, and x_3 = depth of each media (in); P_1,P_2, and P_3 = price of each media ($/cu ft); t = time units of filtration period (15 min); W = backwash cost ($/sq ft); the 43,200 represents the number of minutes in a 30-day month; and the 10^6 converts to million gallons.

Eliminating the A in the numerator and denominator and changing to posynomial form, Equation 5.3 becomes

$$(1.92)(Rx_1P_1Q^{-1} + Rx_2P_2Q^{-1} + Rx_3P_3Q^{-1}) + 66{,}700\ Wt^{-1}Q^{-1} \tag{5.4}$$

The backwash cost, W, is equal to the product of the backwash rate, the backwash time, and the cost of backwash water.

There are several restrictions on the objective function's decision variables, Q, t, x_1, x_2, and x_3. These restrictions can also be expressed in mathematical equations forming the constraints of the model.

Constraints

Before the constraints can be established, equations for the filter performance curves must first be developed. The two performance curves were previously described in Chapter IV and shown there as Figures 4.1 and 4.2. The concave curves can each be approximated by two straight line segments. An abscissa value of 0.9 has been selected for the break point for the straight line segments of each curve thereby establishing a uniform range of values for each set of performance equations.

The bottom portion of the curve in Figure 4.1 (abscissa $\leq$ .9) has a slope of 2.36 while the top has a slope of 1.70. The line intercept is 0.01 at the ordinate and 0.03 at the abscissa. The equation of the bottom portion of the curve is therefore

$$\text{Log}\,\frac{U}{L} = \text{Log}\,(0.1) + 2.36\,\text{Log}\left[\frac{(Q)^{.2}(d_e)^{.3}(t)}{(L)^{.9}}\right] - \text{Log}\,0.3 \qquad (5.5)$$

Taking the antilog

$$\frac{U}{L} = .01 \left[\frac{(Q)^{.2}(d_e)^{.3}t}{.3(L)^{.9}}\right]^{2.36}$$

$$= .172\ (Q)^{.47}\ (d_e)^{.71}\ (t)^{2.36}\ (L)^{-2.12} \qquad (5.6)$$

In the geometric inequality form

$$5.82\ (U)\ (Q)^{-.47}(d_e)^{-.71}\ (t)^{-2.36}\ (L)^{1.12} \leq 1 \qquad (5.7)$$

Similarly, for the top portion

$$6.49\ (U)\ (Q)^{-.34}\ (d_e)^{-.51}\ (t)^{-1.70}\ (L)^{.53} \leq 1 \qquad (5.8)$$

For Figure 4.2, the slope of the bottom portion of the curve is 1.39, and the top slope is 1.0. The ordinate intercept is 10^{-4} and the abscissa intercept is 0.22. Thus the equation of the bottom portion of the curve is

$$\text{Log}\left[\frac{(d_e)^{1.4}(H_t - H_o)}{(Q)^{1.2}(C_o)^{.4}(L)^{1.7}}\right] = \text{LOG}\ (10^{-4})$$

$$+\ 1.39\ \text{Log}\left[\frac{(Q)^{.2}(d_e)^{.3}(t)}{(L)^{.9}}\right] - \text{Log}\ (.22) \qquad (5.9)$$

Taking the antilog

$$\frac{d_e^{1.4}(H_t - H_o)}{(Q)^{1.2}(C_o)^{.4}(L)^{1.7}} = 10^{-4} \quad \frac{(Q)^{.2}(d_e)^{.3}(t)^{1.39}}{.22(L)^{.9}} = 8.2 \times 10^{-4}(Q)^{.28}$$

$$(d_e)^{.42}(t)^{1.39}(L)^{-.125} \tag{5.10}$$

or

$$t^{1.39} = \frac{(H_t - H_o)(d_e)^{.98}}{8.2(10^{-4})(Q)^{1.48}(L)^{.45}(C_o)^{.4}} \tag{5.11}$$

Similarly the equation for the top portion of the curve in Figure 4.2 is

$$t = \frac{(H_t - H_o)(d_e)^{1.1}}{7.5(10^{-4})(Q)^{1.4}(L)^{.8}(C_o)^{.4}} \tag{5.12}$$

Equation 5.11 can be expressed in the geometric programming inequality form as

$$8.2(10^{-4})(Q)^{1.48}(L)^{.45}(C_o)^{.4}(d_e)^{-.98}(H)^{-1}(t)^{1.39} \leq 1 \tag{5.13}$$

and Equation 5.12 can be expressed as

$$7.5(10^{-4})(Q)^{1.4}(L)^{.8}(C_o)^{.4}(d_e)^{-1.1}(H)^{-1}(t) \leq 1 \tag{5.14}$$

where $H = H_t - H_o$ or $H + H_o = H_t$. Thus

$$H H_t^{-1} + H_o H_t^{-1} \leq 1 \tag{5.15}$$

For a minimum length of filter run, t_{min}

$$t_{min} t^{-1} \leq 1 \tag{5.16}$$

Equations 5.8 and 5.14 will be used initially for the filter performance constraints. After the model is optimized, the abscissa value of $(Q)^{.2}(d_e)^{.3}(t/L)^{.9}$ will be calculated, and, if this value is less than or equal to 0.9, Equations 5.7 and 5.13 will be substituted for Equations 5.8 and 5.14 and the model reoptimized. Equations 5.8, 5.14, 5.15, and 5.16 are thus the first three constraints of the model. The variable d_e is calculated using Equation 5.13 as $d_e = X_1 d_1 + X_2 d_2 + X_3 d_3$ where d_i is the diameter size of media i and X_i is the percentage of the media by volume. The bed depth L is also a sum of the individual media layers, $x_i = x_i/L$.

$$x_1 L^{-1} + x_2 L^{-1} + x_3 L^{-1} = 1 \tag{5.17}$$

$$d_1 x_1 d_e^{-1} L^{-1} + d_2 x_2 d_e^{-1} L^{-1} + d_3 x_3 d_e^{-1} L^{-1} = 1 \tag{5.18}$$

Equations 5.17 and 5.18 must equal unity. If they are to be used in a geometric programming algorithm of inequalities, each will require two equations with one ≤ 1 and the other ≥ 1.

Since the filter is to be installed in an existing bed, headloss must not exceed the available clearance, Y. The clearance is measured from the top of the filter walls down to the surface of the course gravel which is used in most miltimedia filters to support the third media layer of fine garnet. That is

$$12H_t Y^{-1} \leq 1 \tag{5.19}$$

Another clearance restriction involves the wash water troughs. The expanded bed depth during backwash must not exceed this clearance or media will be lost with the wash water. The amount of expansion of each media during backwashing depends on the specific gravity of the media and the backwash rate. During the pilot plant studies (discussed in Chapter IV), an optimum backwash rate can be established and the percentage of expansion E_i determined for each type of media. The total expanded depth must not exceed the available clearance, Y_1, of the wash-water troughs

$$x_1 Y_1^{-1} (1 + E_1) + x_2 Y_1^{-1}(1 + E_2) + x_3 Y_1^{-1}(1 + E_3) \leq 1 \tag{5.20}$$

To ensure that production exceeds expected demand, a minimum filter flow, G, in gallons per day (gpd) is established. This minimum production in turn sets a minimum on the filter flow rate with an allowance made for lost production time during backwash. It is estimated that at most four percent of the 1440 minutes in a day will be used for backwash. This percentage is based on three backwashes

per day with 15 to 20 minutes for each wash.

$$(.96)1440\ Q\ A \geq G \tag{5.21}$$

or

$$.0007236\ Q^{-1}A^{-1}G \leq 1 \tag{5.22}$$

A minimum bed depth is required to prevent unfiltered water from breaking through the bed. Hudson [28] concluded that breakthrough will occur when

$$\frac{Q\ d^3 H_t}{L} = B\ \frac{60}{T + 10} \tag{5.23}$$

in which, T = water temperature (^{o}F) and B = a breakthrough index which varies for different waters depending on the response to coagulation and degree of pretreatment. Assuming that d_e is the equivalent diameter for the trimedia

$$\frac{T + 10}{60}\ Q\ d_e^3 H_t L^{-1} B^{-1} \leq 1 \tag{5.24}$$

An upper bound d_{max} and a lower bound d_{min} on each media diameter can be established from information furnished by the media supplier. Thus

$$d_i d_{i_{max}}^{-1} \leq 1, \quad i = 1, 2, 3 \tag{5.25}$$

and

$$d_i^{-1} d_{i_{min}} \leq 1 \quad i = 1, 2, 3 \tag{5.26}$$

For proper stratification of the media after backwash, a size ratio of adjacent media must be established. This ratio can be determined satisfactorily by the principle of equal settling [39]. The ratio has been calculated by Conley and Hsiung [8] as

$$\frac{d_i}{d_{i+1}} \leq \frac{(D_{i+1} - D_{mi})}{(D_i - D_m)} \frac{1}{1.6} \quad , \quad i = 1, 2 \tag{5.27}$$

or

$$\frac{1.6\, d_i (D_i - D_{mi})}{d_{i+1}(D_{i+1} - D_{mi})} \leq 1 \quad , \quad i = 1, 2 \tag{5.28}$$

where

$$D_{mi} = e_i D + 1/2(D_i + D_{i+1})(1 - e_i) \quad , \quad i = 1, 2 \tag{5.29}$$

in which, e_i = average porosity of the mixing interface (two adjacent media) of the expanded bed near the end of backwash, D_i and D_{i+1} = density of media to be compared, and D_{mi} = density of mixture.

Other constraints could possibly be developed on an individual basis for a specific design problem, but those presented in this

section are generally most restrictive on the decision variables.

An Optimum Solution

Equation 5.4 represents the objective function and Equations 5.8, 5.14, 5.15, 5.16, 5.17, 5.18, 5.19, 5.20, 5.22, 5.24, 5.25, 5.26, and 5.28 describe the constraints of the mathematical model.

A solution to the model can be found by completing the following steps:

1. Find values for the constants in the model.
2. Determine coefficients for each term in the objective function and the constraints and set up the mathematical form of the problem.
3. Optimize the model using GPS or some alternate procedure.
4. Using the results of the decision variables determined by step 2, calculate $Q^{.2}d_e^{.3}t/L^{.9}$. If this value is less than or equal to 0.9, use Equations 5.7 and 5.13 for the first two constraints and return to step 2 above.

A sample set of constants will be identified in the following chapter, and an optimization will be performed to test the model.

CHAPTER VI

A SAMPLE PROBLEM

Constant Values

The first step in the problem solution is to identify the constant values. These values can be obtained from plant records, pilot studies, and design criteria for the filter bed where the trimedia is to be installed. The values listed below are representative of those for a typical plant. The source of the value is shown where applicable:

i = interest rate = 5 percent

n = number of periods = 20 years

A = surface area of filter = 242 sq. ft.

P_1 = price of anthracite [46] = \$0.60/cu. ft.

P_2 = price of sand = \$0.20/cu. ft.

P_3 = price of garnet [1] = \$3.00/cu. ft.

Backwash rate [7,43] = 15 gpm/sq. ft.

Length of backwash [7] = 6.7 min.

Price of backwash water = \$0.35/thousand gallons

C_o = influent turbidity = 20 Jackson Turbidity Units (JTU)

C = effluent turbidity = 0.2 JTU

H_o = initial headloss = 1.333 ft.

t_{min} = minimum length of filter run = 40 periods (each period is 15 minutes or 10 hours total)

d_1 = diameter of anthracite [7] = 1.0-2.0 mm

d_2 = diameter of sand [7] = 0.4-0.8 mm

d_3 = diameter of garnet [7] = 0.2-0.4 mm

Y = clearance of bed = 96 in.

Y_1 = clearance of wash troughs = 45 in.

E_1 = expansion of anthracite [48] = 41 percent

E_2 = expansion of sand [48] = 35 percent

E_3 = expansion of garnet [48] = 10 percent

G = minimum daily production = 1,000,000 gpd

B = breakthrough index [2] = 1 (would equal $1x10^{-3}$ if d_e was expressed in cm)

T = water temperature = 50°F

U = deposit index [3] (equal to the value of the chi-square variable at the C/C_o percentage point) = 22

e_1 = average porosity for coal-sand-water mixture [8] = 0.85

e_2 = average porosity for sand-garnet-water mixture [8] = 0.80

D_1 = specific gravity of anthracite [7] = 1.5

D_2 = specific gravity of sand [7] = 2.6

D_3 = specific gravity of garnet [7] = 3.8

D = specific gravity of water = 1.0

The media diameters will be established as constants to reduce the number of variables and constraints in the model. The use of these three constants eliminates three variables and eight constraints. To establish the diameters, the density of the anthracite-sand-water and sand-garnet-water mixtures must first be calculated per Equation 5.27.

$D_{m1} = .85(1.0) + 1/2(1.5 + 2.6)(1 - .85) = 1.157$

$D_{m2} = .80(1.0) + 1/2(2.6 + 3.8)(1 - .80) = 1.44$

The diameter ratio of adjacent media is calculated using Equation 5.28

$$\frac{1.6\ d_1(1.5 - 1.157)}{(2.6 - 1.157)} \leq d_2$$

$$.3803\ d_1 \leq d_2 \qquad (6.1)$$

$$\frac{1.6\ d_2(2.6 - 1.44)}{(3.8 - 1.44)} \leq d_3$$

$$.7864\ d_2 \leq d_3 \qquad (6.2)$$

If the diameter of garnet, d_3, is set at 0.4mm, then d_1 and d_2 can be set per Equations 6.1 and 6.2 at 1.337 and .5086 respectively.

$$R = \frac{.05\ (1 + .05/12)^{240}}{(1 + .05/12)^{240} - 1} = .0065701$$

and the backwash cost is

$$W = 15(6.7)(.00035) = \$\ .035175/\text{sq ft}$$

With the constants identified, step two is to determine the coefficients of the variables and set up the equations of the model.

Model for Sample Problem

The mathematical programming formulation of the reduced problem is given below:

Minimize

$$.00656\ x_1Q^{-1} + .002189\ x_2Q^{-1} + .03284\ x_3Q^{-1} + 2345\ t^{-1}Q^{-1}$$

Subject to

$$142.86\ Q^{-.34}t^{-1.7}d_e^{-.51}L^{.53} \leq 1$$

$$.00248\ Q^{1.4}L^{.8}d_e^{-1.1}H^{-1}t \leq 1$$

$$H\ H_t^{-1} + 1.333\ H_t^{-1} \leq 1$$

$$40\ t^{-1} \leq 1$$

$$x_1L^{-1} + x_2L^{-1} + x_3L^{-1} = 1$$

$$1.337\ x_1d_e^{-1}L^{-1} + .5086\ x_2d_e^{-1}L^{-1} + .4\ x_3d_e^{-1}L = 1$$

$$.125\ H_t \leq 1$$

$$.03133\ x_1 + .03x_2 + .02444x_3 \leq 1$$

$$3.0\ Q^{-1} \leq 1$$

$$Q\ d_e^3H_tL^{-1} \leq 1$$

The model now has nine variables, ten constraints, and 21 terms. As a geometric programming problem it thus has 21 - (9+1) = 11 degrees of difficulty. This would be increased to 17 degrees if two inequalities were required for each of the two equality constraints. Two computer programs were tested on this problem but both experienced great difficulty in finding a solution. Computer programs for solving geometric programming

problems with large numbers of degrees of difficulty have as yet not been perfected [49].

To obtain an efficient solution (considering computational difficulty), the model was separated into two parts. The first part uses geometric programming to select the optimal flow rate, length of filter run, total depth of media, equivalent grain diameter, and head. The second part then uses the results from part one and a linear programming algorithm to compute the depth of anthracite, sand, and garnet. This approach makes use of the strong points of each technique. Successive application of more than one optimization technique in problem solution is becoming increasingly popular in solving complex problems.

Geometric Programming Solution

The geometric programming problem is

Minimize

$$2345\ t^{-1}Q^{-1}$$

Subject to

$$142.86\ Q^{-.34}t^{-1.7}d_e^{-.51}L^{.53} \leq 1$$

$$.00248\ Q^{1.4}L^{.8}d_e^{-1.1}H^{-1}t \leq 1$$

$$7.5\ H_t^{-1} \leq 1$$

$$.162\ H \leq 1$$

$$40\ t^{-1} \leq 1$$

$$Q\ d_e^3\ H_t L^{-1} \leq 1$$

A lower bound of 7.5 feet and an upper bound of 6.17 feet were established on H_t and H in constraints three and four above in lieu of using the third and seventh constraints of the model. The lower bound on Q was also omitted. These actions avoided two terms and two degrees of difficulty. The problem above has seven terms, six variables, and zero degrees of difficulty. Application of the Edwards Geometric Programming Systems yields the following solution.

$Q = 3.0$ gpm/sq ft

$t = 40.0$ time periods (10 hours)

$L = 25.0$ inches

$d_e = 1.025$ mm

$H = 6.17$ ft

$H_t = 7.5$ ft

Value of objective function = \$18.9302/million gal.

Linear Programming Solution

The linear problem consists of the x_1, x_2, and x_3 terms from the model plus the following bounds on each media to ensure a proper mixture.

$$0.6L \leq x_1 \leq 0.75L$$

$$0.2L \leq x_2 \leq 0.3L$$

$$0.05L \leq x_3$$

The resulting linear programming problem is

Minimize

$$.002187\ x_1 + .00073\ x_2 + .01095\ x_3$$

Subject to

$$1.337\ x_1 + .5086\ x_2 + .4\ x_3 = 25.625$$

$$x_1 + x_2 + x_3 = 25$$

$$.03133\ x_1 + .03\ x_2 + .02444\ x_3 \leq 1$$

$$x_1 \leq 18.75$$

$$x_1 \geq 15$$

$$x_2 \leq 7.5$$

$$x_2 \geq 5$$

$$x_3 \geq 1.25$$

The above problem was solved using MPS/360 yielding

x_1 = 15.8 in

x_2 = 7.5 in

x_3 = 1.7 in

Value of objective function = $ .0586/million gallons

Summary of Results

The total of the objective function for the model is 18.9302 + .0586 = $18.9888/million gallons. The abscissa and ordinate values of the performance curves were calculated to test the

accuracy of the performance equations yielding

$$\frac{Q^{.2}\, d_e^{.3}\, t}{L^{.9}} = 2.77$$

$$\frac{U}{L} = 0.87$$

$$\frac{d_e^{1.4}\,(H_t - H_o)}{Q^{1.2} C_o^{.4} L^{1.7}} = .0022$$

The above results are very close to the values shown on the performance curves.

The influent concentration, C_o, and grain diameter, d_i, were varied for other runs of the model. Results of one of these runs (run 2) is compared in Table 6.1 with the previous data from run 1.

	Run 1 $C_o = 20$ $d_1 = 1.337$ $d_2 = .5086$ $d_3 = .4$	Run 2 $C_o = 5$ $d_1 = 1.05$ $d_2 = .4$ $d_3 =$ none
Q	3.0	4.75
t	40.0	40.0
L	25.0	15.7
d_e	1.025	0.76
H	6.17	6.17
H_t	7.5	7.5
x_1	15.8	8.7
x_2	7.5	7.5
x_3	1.7	none
Obj. Function	18.9888	12.3409
$Q^{.2}\ d_e^{.3}\ t/L^{.9}$	2.77	4.22
U/L	.87	1.78
$\frac{d_e^{1.4}(H_t - H_o)}{Q^{1.2}\ C_o^{.4} L^{1.7}}$	.0022	.0032

Table 6.1 Output Comparisons

CHAPTER VII

CONCLUSIONS AND RECOMMENDATIONS

Filter Cost Model

Mathematical cost modeling is a practical approach to filter design. Such models are of course dependent on the availability of curves or equations that can predict filter performance. Hsiung, as indicated previously, has provided a method for developing the needed curves into a cost model. The cost model and optimization solution methodology presented in this report will also be useful in the determination of the optimum operational policies for an existing filtration plant.

Refinements to the model are encouraged. Filtration is a highly complex process and there are no other items and constraints that could be added to the model to make it more universal. This report merely introduces the technique in the hope that experienced environmental design engineers can improve on its form and find areas where it can be practically applied.

Geometric Programming Optimization

It is reasonable to expect the filter performance equations to be as complex as the process they are describing. Without geometric programming, linear or quadratic approximations to the equations would have to be made before a solution can be obtained. Geometric programming, however, can be used to solve these highly nonlinear equations and provide accurate answers. Also, very little additional computations are

necessary to determine the new value of the objective function whenever prices or cost coefficients change.

Geometric programming would be even more valuable if a general computer algorithm can be developed to solve problems with many degrees of difficulty. The Edwards GPS is a good basis for such a program, but it sorely needs a simplex subroutine to find an initial feasible solution in the linear dual space. Without this initial point, it experiences difficulty at times locating the feasible region. There are several search routines that could be used for the degree of difficulty problems, but one must be careful in compromising between efficiency and level of significance. General search programs with many significant figures are desirable, but such programs can consume a great deal of computer time.

Combining Optimization Techniques

Arguments might be offered that the separation of the problem into two subproblems does not guarantee a global optimum. These arguments probably have some basis, but the global optimum is an aspiration that few design engineers actually achieve. In modeling, it is usually necessary to loop between deductive tractability and assumptions for characterization.

The combination of operations research techniques can have wide application and be extremely useful in the solution of real world problems. Combining geometric programming with linear programming, dynamic programming, search, simulation, etc. can easily lead to the solution of many complex problems.

Future Extensions

The model developed in this research needs to be tested with actual data and experiments run to compare predictions with experimental results. A sensitivity analysis also needs to be performed to measure the effect of each variable and constant on the final results.

The modeling technique itself can be used to predict performance of wastewater and diatomaceous earth filters. Filtration theory for diatomaceous earth filters is much more refined than that for sand or multimedia filters, and a generally accepted set of performance equations are available [14]. The model can also be extended in water filtration to include more of the costs involved in the construction of a new filtration plant.

The geometric programming technique has a wide range of potential applications in environmental engineering including hydrology, chemical treatment processes, water resource analysis, and others.

Mathematical modeling is an inexpensive method of describing and testing a system. It can help the engineer to build conceptual structures out of perceptual confusion.

LIST OF REFERENCES

1. BARTON MINES SALES BROCHURE, North Creek, N. Y., 1968.

2. Beightler, C. S., Crisp, R. M., and Meier, W. L., "Optimization by Geometric Programming," J. INDUS. ENG., Vol. 19, No. 3, March 1968.

3. Beyer, W. H., HANDBOOK OF TABLES FOR PROBABILITY AND STATISTICS, 2nd. Ed., The Chemical Rubber Co., Cleveland, Ohio, 1968.

4. Camp, T. R., "Theory of Water Filtration," PROC. ASCE, J. SAN. ENG. DIV., Vol. 90, No. SA4, 1964.

5. Charnes, A., and W. W. Cooper, "Optimizing Engineering Design Under Inequality Constraints," NORTHWESTERN UNIV. ONR RES. MEMO 64, Evanston, Ill., August, 1962.

6. Conley, W. R., "Experience with Anthracite Sand Filters," J. AMER. WATER WORKS ASSOC., Vol. 53, 1961.

7. Conley, W. R., Personal Correspondence, October 1970.

8. Conley, W. R., and Hsiung, K., "Design and Application of Multimedia Filters," J. AMER. WATER WORKS ASSOC., Vol. 61, 1969.

9. Craft, T. F., "Review of Sand Filtration Theory," J. AMER. WATER WORKS ASSOC., Vol. 58, 1966.

10. Deb, A. K., "Numerical Solution of Filtration Equations," PROC. ASCE, J. SAN. ENG. DIV., Vol. 96, No. SA2, 1970.

11. Deb, A. K., "Theory of Sand Filtration," PROC ASCE, J. SAN. ENG. DIV., Vol. 95, No. SA3, 1969.

12. DeGarmo, E. P., ENGINEERING ECONOMY, 4th Ed., The MacMillan Co., London, 1967.

13. Diaper, E. W. J., and Ives, K. J., "Filtration Through Size Grained Media," PROC. ASCE, J. SAN ENG. DIV., Vol. 91, No. SA3, 1965.

14. Dillingham, J. H., Cleasby, J. L., and Bauman, E. R., "Optimum Design and Operation of Diatomite Filtration Plants," J. AMER. WATER WORKS ASSOC., Vol. 58, 1966.

15. Dostal, K. A., Harrington, J. J., Clark, R. M., and Robeck, G. G., "Development of Optimization Models for Carbon Bed Absorption," J. AMER. WATER WORKS ASSOC., Vol 58, 1966.

16. Duffin, R. J., "A Technique for Optimizing Engineering Designs, WESTINGHOUSE R AND D LETTER, Vol. 7, April, 1964.

17. Duffin, R. J.,"Cost Minimization Problems Treated by Geometric Means," OPNS. RES., Vol. 10, No. 5, 1962.

18. Duffin, R. J., and E. L. Peterson, "Constrained Minimum Treated by Geometric Means," WESTINGHOUSE SCIENTIFIC PAPER 84-158-129-P3, Pittsburgh, March 1964.

19. Duffin, Richard J., Peterson, Elmor L., and Zener, Clarence M., GEOMETRIC PROGRAMMING, John Wiley & Sons, Inc., New York, 1967.

20. Edwards, T. H., "A Computer Program to Solve the Generalized Geometric Programming Problem," M. S. Thesis, Texas A&M University, College Station, Texas, 1969.

21. Fair, G. M., Geyer, J. C., and Okun, D. A., WATER AND WASTEWATER ENGINEERING, Vol. 2, John Wiley & Sons, Inc., New York, 1968.

22. Federal Council on Science and Technology, "A Ten-Year Program in Water Resources Research," Executive Office of the President, Washington, D. C., February,1966.

23. Hadley, G., NONLINEAR AND DYNAMIC PROGRAMMING, Addison-Wesley, Reading, Mass., 1964.

24. Hardy, G. H., J. E. Littlewood, and G. Polya, INEQUALITIES, Cambridge Univ. Press, New York, 1959.

25. Hooke, R., and T. A. Jeeves, "'Direct Search' Solution of Numerical and Statistical Problems," J. ASSN. COMP. MACH., Vol. 8, No. 2, April, 1961.

26. Hsiung, K., "Prediction of Performance of Granular Filters for Water Treatment," Ph.D. Dissertation, Iowa State University, Ames, Iowa, 1967.

27. Hsiung, K., and Cleasby, J. L., "Prediction of Filter Performance," PROC. ASCE, J. SAN. ENG. DIV., Vol. 94, No. SA6, 1968.

28. Hudson, Jr., H. E., "Declining Rate Filtration," J. AMER. WATER WORKS ASSOC., Vol. 51, 1959.

29. Ives, K. J., "New Concepts in Filtration," WATER AND WASTE ENG., Vol 65, 1961.

30. Ives, K. J., "Progress in Filtration," J. AMER. WATER WORKS ASSOC., Vol. 56, 1964.

31. Ives, K. J., "Rational Design of Filters," PROC. INST. CIVIL ENGRS., Vol 16, 1960.

32. Ives, K. J., "Simplified Rational Analysis of Filter Behavior," PROC. INST. CIVIL ENGRS., Vol 25, 1963.

33. Iwasaki, T. Some Notes on Sand Filtration. J. AMER. WATER WORKS ASSOC., Vol. 29, 1937.

34. Kozeny, J., WASSERKRAFT AND WASSERWIRTSCHAFT, Vol. 22, No. 67, 1927.

35. Kuhn, H. W., and Tucker, A. W., "Nonlinear Programming," PROC. SECOND BERKELEY SYMPOSIUM ON MATH. STAT. AND PROB., 1950, Univ. of Calif. Press, 1951.

36. Laughlin, J. E. and DuVall, T. E., "Multiple Media Filters are Key to Texas Plant Expansion," WATER AND WASTES ENGINEERING, Vol. 5, 1968.

37. Letto, A. R., "A Computer Program for Optimization Using Pattern Search and Gradient Summation Techniques," Unpublished Masters Paper, Texas A&M University, Department of Industrial Engineering, College Station, Texas, 1968.

38. Maass, Arthur, et al., DESIGN OF WATER RESOURCES SYSTEMS, Harvard University Press, Cambridge, 1962.

39. McCabe, W. L. and Smith, J. C., UNIT OPERATIONS OF CHEMICAL ENGINEERING, McGraw-Hill Book Co. Inc., New York, 1956.

40. Meier, W. L., and Beightler, C. S., "An Optimization Method for Branching Water Resources Systems," WATER RESOURCES RES., Vol. 3, No. 3, 1967.

41. Meier, W. L., "Optimal Planning of Nonserial Multistage Water Resources Systems," Doctoral Dissertation, The University of Texas, Austin, 1967.

42. Mintz, D. M., Report to International Water Supply Association, Seventh Congress, Barcelona, 1966.

43. Mueller, Jr., H. M., "High Rate Sedimentation and Filtration," WATER AND WASTES ENGINEERING, Vol. 7, No. 6, 1970.

44. O'Melia, C. R. and Struum, W., "Theory of Water Filtration," J. AMER. WATER WORKS ASSOC., Vol. 59, 1967.

45. Ott, C. R. and Bogan, R. H., "Theoretical Evaluation of Filtering Modeling Experiments," PROC. ASCE, JOUR. SAN. ENG. DIV., Vol. 96, No. SA2, 1970.

46. Palmer Anthrafilt Equipment Co., ANTHRAFILT PRICE SCHEDULE R. P. 62, Fairview, Pa., 1964.

47. Passy, U., and Wilde, D. J., "Generalized Polynomial Optimization," SIAM J. OF INDUS. AND APPL. MATH., Vol. 15, No. 5, September, 1967.

48. Rimer, A. E., "Filtration Through a Trimedia Filter," PROC. ASCE, JOUR. SAN. ENG. DIV., Vol. 94, No. SA3, 1968.

49. Scherfig, J., Schinzinger, R., and Morgan, T. W., "Waste Treatment Optimization by Geometric Programming," Paper Presented at Water Pollution Research Conference, San Francisco, July-August 1970.

50. Sedmore, K. J., "We Converted to High Rate Filtration," AMERICAN CITY, Vol. 83, 1968.

51. Sholji, I., "The Filtration of Suspensions Through Deep Granular Filters," Ph.D. Thesis, University College, London, 1963.

52. Stein, P. C., A STUDY OF THE THEORY OF RAPID FILTRATION OF WATER THROUGH SAND, Ph.D. Dissertation, MIT, Cambridge, Mass., 1940.

53. Thomas, H. A., and Revelle, Roger, "On the Efficient Use of High Aswan Dam for Hydropower and Irrigation," MANAGEMENT SCI., Vol. 12, No. 8, 1965.

54. United States Congress, "Water Resources Planning Act," Public Law 89-80, 89th Congress, S. 21, July 22, 1965.

55. Wilde, D. J., "A Unified Approach to Multivariable Optimization Theory, INDUS. ENGNG, AND CHEM., Vol. 57, August, 1965.

56. Wilde, D. J., and Beightler, C. S., FOUNDATIONS OF OPTIMIZATION, Prentice-Hall, Englewood Cliffs, N. J., 1967.

57. Zener, C., "A Further Mathematical Aid in Optimizing Engineering Designs," PROC. NATL. ACAD. OF SCI., USA, Vol. 48, 1962.

58. Zener, C., "A Mathematical Aid in Optimizing Engineering Designs," PROC. NATL. ACAD. OF SCI., USA, Vol. 47, No. 4, April, 1961.

59. Zener, C., "Minimization of System Costs in Terms of Subsystem Costs," PROC. NATL. ACAD. OF SCI., USA, Vol. 51, 1964.

60. Zener, C., and Duffin, Richard J., "Optimization of Engineering Problems," WESTINGHOUSE ENGINEER, September, 1964.

APPENDIX A

PROGRAM LISTING

```
C.....                                                                  00001
C.....                                                                  00002
C.....                                                                  00003
C.....        * * *  GEOMETRIC PROGRAMMING SYSTEM   * * *               00004
C.....                                                                  00005
C.....                                                                  00006
C..... THIS ALGORITHM IS DESIGNED TO MINIMIZE GENERALIZED POLYNOMIALS   00007
C..... BY THE METHOD OF GEOMETRIC PROGRAMMING.  FOR THE CONVEX CASE,    00008
C..... (ALL COEFFICIENTS POSITIVE),IT GUARANTEES A GLOBAL MINIMUM.      00009
C..... FOR THE NONCONVEX CASE, ONE PSEUDO-OPTIMUM POINT IS LOCATED      00010
C..... WHICH MAY OR MAY NOT BE THE GLOBAL MINIMUM.                      00011
C..... WHEN DEGREES OF FREEDOM ARE PRESENT IN THE DUAL SOLUTION A       00012
C..... PATTERN SEARCH  ALGORITHM BY A.R. LETTO IS EMPLOYED TO LOCATE    00013
C..... A DUAL OPTIMUM.                                                  00014
C.....                                                                  00015
C.....                                                                  00016
C.....                                                                  00017
      IMPLICIT REAL*8(A-B,D-H,O-Q,U-Z),  INTEGER(C,R-T)                 00018
      REAL*8 NOTE(10)                                                   00019
      COMMON W(41),XCFSC,XCASS,ADAF,XLLIM,KC,IFEAS,ISTOP,LACT,MPT,IDDIF,00020
     1LPRINT                                                            00021
      COMMON/GUS/X(20),XBASE(20),XTEMP(20),XTEST(20),PARTZ(20),PARTH(20)00022
      COMMON/TOM/A(10,10,10),B(31,32),XCON(10,10),SIGMA(10,10),SIG(10), 00023
     1ROW(32),COL(32),TSUBM(10),CAPM,CAPN,CAPNP1,CAPNP2,CAPT,CAPTP1     00024
C.....                                                                  00025
C..... CAPM = NUMBER OF ROWS (OBJECTIVE FUNCTION AND ALL CONSTRAINTS)   00026
C..... CAPN = NUMBER OF VARIABLES                                       00027
C..... TSUBM(M) = NUMBER OF TERMS IN THE M-TH ROW                       00028
C..... CAPM, CAPN, TSUBM(M) MUST ALL BE LESS THAN OR EQUAL TO 10.       00029
C..... SIG(1) (=+ OR -1) = SIGN OF OBJECTIVE FUNCTION AT MINIMUM.
```

```
C.......SIG(M), M = 2,...CAPM, (=+ OR -1) = UPPER BOUND ON CONSTRAINT M.
C.......
C * * *  E N T E R I N G   F L O W C H A R T   B L O C K   A   * * * *
  100 READ (5,1) CAPM,CAPN,(TSUBM(M),M=1,CAPM),(SIG(M),M=1,CAPM)
    1 FORMAT (22I2)
C.......
C....... CLOCK IS NOT AVAILABLE IN WATFOR.
C.......
   92 CALL CLOCK(HRS,AMIN,ASEC)
C * * *  E N T E R I N G   F L O W C H A R T   B L O C K   B   * * * *
      WRITE(6,30)
   30 FORMAT('1'T43,'***POLYNOMIAL MINIMIZATION BY GEOMETRIC PROGRAMMING
     1***'/)
C.......
C....... READ-IN / READ-OUT TITLE.
C.......
      READ (5,102) (NOTE(I),I=1,10)
  102 FORMAT(10A8)
      WRITE(6,110) (NOTE(I),I=1,10)
  110 FORMAT(T40,10A8)
C.......
C....... READ LOWER LIMIT FOR STEP SIZE IN LETTO PATTERN SEARCH
C....... SUBROUTINE.  DEFAULT OPTION/XLLIM = 10.**(-6)
C....... READ LPRINT, AN INTEGER FROM 0 TO 4.  HIGHER VALUES CAUSE
C....... LESS INFORMATION ON THE PATTERN SEARCH TO BE PRINTED OUT.
C.......
      READ(5,210) XLLIM,LPRINT
  210 FORMAT(E10.0,I1)
      IF(XLLIM.LE.0.0) XLLIM = 10.**(-6)
      I = CAPM-1
```

```
      WRITE (6,103)CAPN,I                                                    00060
  103 FORMAT(////T40,'PRIMAL PROGRAM.'/T40,15('=')////                       00061
     1T40,'THE NUMBER OF PRIMAL VARIABLES = 'I2/                             00062
     1T40,'THE NUMBER OF PRIMAL CONSTRAINTS = 'I2)                           00063
      IF(CAPN.LE.0.OR.CAPM.LE.0)  CALL BLOWUP(1,&104)                        00064
      IF(CAPN.GT.10.OR.CAPM.GT.10) CALL BLOWUP(2,&104)                       00065
      WRITE(6,35)                                                            00066
   35 FORMAT(///T55,'*** GENERALIZED POLYNOMIAL ***'///T40,'ROW'             00067
     1T46,'TERM'T54,'COEFFICIENT'T69,'PRIMAL'T81,'EXPONENT'T94,'SIGNUM'/     00068
     1T56,'OF TERM'T68,'VARIABLE'T80,'OF VARIABLE'T94,'OF ROW'/              00069
     1 T40,60('-')/T40,'MINIMIZE0'/)                                         00070
      CAPT = 0                                                               00071
      CAPNP1 = CAPN+1                                                        00072
      CAPNP2 = CAPN+2                                                        00073
      DO 8 M=1,CAPM                                                          00074
      IF(IABS(SIG(M)).NE.1.OR.TSUBM(M).LE.0) CALL BLOWUP(1,&104)             00075
      IF(TSUBM(M).GT.10) CALL BLOWUP(2,&104)                                 00076
      I =TSUBM(M)                                                            00077
      DO 4 T=1,I                                                             00078
      CAPT = CAPT+1                                                          00079
C.......                                                                     00080
C.......  READ XCON, THE COEFFICIENT OF THE T-TH TERM AND THE M-TH ROW.      00081
C.......                                                                     00082
      READ(5,2) XCON(M,T)                                                    00083
    2 FORMAT(E25.0)                                                          00084
      WRITE (6,36) M,T,XCON(M,T)                                             00085
   36 FORMAT(/T40,I2,T47,I2,T51,G15.8)                                       00086
      IF(XCON(M,T).EQ.0.0) CALL BLOWUP(1,&104)                               00087
      XABS =DABS(XCON(M,T))                                                  00088
      SIGMA(M,T) = XCON(M,T)/XABS                                            00089
```

```
      XCON(M,T) = XABS                                                  00090
                                                                        00091
C.......                                                                00092
C....... READ A(M,T,N), THE EXPONENT OF THE N-TH VARIABLE IN THE T-TH   00093
C....... TERM OF THE M-TH ROW OF THE GENERALIZED POLYNOMIAL. IF LAST=1,  00094
C....... N IS CONSIDERED TO BE THE LAST VARIABLE IN THAT TERM.          00095
C.......                                                                00096
    5 READ(5,3) N,A(M,T,N),LAST                                         00097
    3 FORMAT (I2,E16.0,I2)                                              00098
      WRITE (6,37) N,A(M,T,N)                                           00099
   37 FORMAT('+'T67,'* X(',I2,') ** ',G15.8/)                           00100
      IF(N.LE.0.OR.N.GT.CAPN.OR.A(M,T,N).EQ.0) CALL BLOWUP(1,&104)      00101
      IF(LAST.EQ.0) GO TO 5                                             00102
      IF(T.NE.I) WRITE(6,301)                                           00103
  301 FORMAT(T62,'.PLUS.'/)                                             00104
    4 CONTINUE                                                          00105
      IF(M.EQ.1.AND.SIG(1).GT.0) WRITE(6,32)                            00106
   32 FORMAT(T40,'SIGN OF PRIMAL FUNCTION AT MINIMUM ASSUMED TO BE POSIT 00107
     1IVE',/T40,60('-'))                                                00108
      IF(M.EQ.1.AND.SIG(1).LT.0) WRITE(6,132)                           00109
  132 FORMAT(T40,'SIGN OF PRIMAL FUNCTION AT MINIMUM ASSUMED TO BE NEGAT 00110
     1IVE',/T40,60('-'))                                                00111
      IF(M.EQ.1.AND.CAPM.GT.1) WRITE (6,31)                             00112
   31 FORMAT(T40,'SUBJECT TO')                                          00113
      IF(M.GT.1) WRITE(6,33) SIG(M)                                     00114
   33 FORMAT('+'T92,'.LE.'T98,I2/T40,60('-'))                           00115
    8 CONTINUE                                                          00116
C.......                                                                00117
C....... THE NUMBER OF DEGREES OF DIFFICULTY IS THE NUMBER OF TERMS IN  00118
C....... THE OBJECTIVE FUNCTION AND ALL CONSTRAINTS MINUS THE NUMBER    00119
C....... OF PRIMAL VARIABLES MINUS ONE, AND MAY NOT BE LESS THAT ZERO
```

```
C....... OR GREATER THAN TWENTY.                                        00120
C.......                                                                00121
C.......                                                                00122
      IDDIF = CAPT-CAPN-1                                               00123
      WRITE(6,34) CAPT,IDDIF                                            00124
   34 FORMAT(///T40,'THE TOTAL NUMBER OF TERMS = 'I2/                   00125
     1  T40,'THE NUMBER OF ''DEGREES OF DIFFICULTY'' = 'I2)             00126
      IF(IDDIF.LT.0.OR.IDDIF.GT.20) CALL BLOWUP(3,&104)                 00127
C.......                                                                00128
C....... BEGIN CONSTRUCTION OF MATRIX OF COEFFICIENTS OF DUAL           00129
C....... VARIABLES.                                                     00130
C.......                                                                00131
C * * *  E N T E R I N G   F L O W C H A R T   B L O C K   C  * * * * *  00132
      CAPTP1 = CAPT+1                                                   00133
   94 IT=TSUBM(1)                                                       00134
      DO 6 T=1,IT                                                       00135
    6 B(1,T)=SIGMA(1,T)                                                 00136
      IT = IT + 1                                                       00137
      DO 316 T = IT,CAPTP1                                              00138
  316 B(1,T) = 0.0                                                      00139
      TT = 0                                                            00140
      DO 7 M=1,CAPM                                                     00141
      IT = TSUBM(M)                                                     00142
      DO 7 T=1,IT                                                       00143
      TT = TT+1                                                         00144
      DO 7 N=1,CAPN                                                     00145
    7 B(N+1,TT) = SIGMA(M,T)*A(M,T,N)                                   00146
      DO 290 I = 2,CAPNP1                                               00147
  290 B(I,CAPTP1) = 0.0                                                 00148
      B(1,CAPTP1) = SIG(1)                                              00149
C.......
```

```
C....... MATRIX OF COEFFICIENTS OF DUAL VARIABLES IS COMPLETED.                  00150
C....... CALL MAT TO ARRANGE IDENTITY MATRIX IN LEFT SQUARE MATRIX OF B.        00151
C.......                                                                        00152
C.......                                                                        00153
C * * *  E N T E R I N G   F L O W C H A R T   B L O C K   D   * * * * *       00154
      CALL MAT(CAPNP1,CAPTP1,&104)                                              00155
C.......                                                                        00156
C....... IF ALL VALUES ON THE RIGHT HAND SIDE ARE NEGATIVE, SIG(1) IS           00157
C....... OF THE WRONG SIGN AND THE MATRIX MUST BE RE-SOLVED WITH                00158
C....... SIG(1) = -SIG(1).                                                      00159
C.......                                                                        00160
C.......                                                                        00161
      DO 93 I=1,CAPNP1                                                          00162
      IF(B(ROW(I),COL(CAPTP1))) 93,93,96                                        00163
   93 CONTINUE                                                                  00164
  193 DO 95 I = 1,32                                                            00165
      ROW(I) = I                                                                00166
   95 COL(I) = I                                                                00167
      SIG(1) = -SIG(1)                                                          00168
      WRITE (6,220)                                                             00169
  220 FORMAT('1'/T40,'SIGN OF DUAL FUNCTION AT MAXIMUM IS NOT THE SAME A        00170
     1S SPECIFIED FOR PRIMAL.'/T40,'SIG(1) = -SIG(1) '/T40,'EXECUTION CO        00171
     1NTINUES')                                                                 00172
      GO TO 94                                                                  00173
   96 DO 500 I = 1,CAPNP1                                                       00174
      DO 500 J = 1,CAPTP1                                                       00175
  500 B(I+11,J) = B(ROW(I),COL(J))                                              00176
      DO 501 I = 1,CAPNP1                                                       00177
      DO 501 J = 1,CAPTP1                                                       00178
  501 B(I,J) = B(I+11,J)                                                        00179
C * * *  E N T E R I N G   F L O W C H A R T   B L O C K   E   * * * * *
      WRITE (6,40) CAPT,CAPNP1
```

```
   40 FORMAT('1'//T40,'DUAL PROGRAM.'/T40,13('=')/T40 ,                    00180
     1'THE NUMBER OF DUAL VARIABLES = 'I2/                                00181
     1 T40,'THE NUMBER OF DUAL EQUATIONS = 'I2////                        00182
     2 T42,'*** EXTENDED MATRIX OF COEFFICIENTS OF DUAL VARIABLES ***'/   00183
     1 T40,60('-'))                                                       00184
      IF(CAPTP1.GT.12) GO TO 303                                          00185
      DO 25 M=1,CAPNP1                                                    00186
   25 WRITE(6,41)(B(M,I),I=1,CAPTP1)                                      00187
   41 FORMAT(/T40,13F7.3/T43,9F7.3)                                       00188
      GO TO 302                                                           00189
  303 DO 304 M = 1,CAPNP1                                                 00190
  304 WRITE(6,305)(B(M,I),I=1,CAPTP1)                                     00191
  305 FORMAT(/T15,16F7.3/T18,16F7.3)                                      00192
C.......                                                                  00193
C....... IF DEGREES OF DIFFICULTY EXIST, PATTERN SEARCH IS CALLED WITH    00194
C....... IDDIF VARIABLES AND NCON CONSTRAINTS.                            00195
C.......                                                                  00196
C.......                                                                  00197
  302 MPT = CAPM+CAPT                                                     00198
      NCON = MPT-1                                                        00199
      IF(IDDIF.GT.0) WRITE(6,42) IDDIF,NCON,XLLIM,LPRINT                  00200
   42 FORMAT(T40,60('-')/'1'T40,'PATTERN SEARCH.'/T40,15('=')////         00201
     1T40,'THE NUMBER OF VARIABLES = '  I2/                              00202
     1 T40 ,'THE NUMBER OF CONSTRAINTS = 'I2/                             00203
     2  T40,'THE LOWER LIMIT ON STEP SIZE = ',1PE12.4/                    00204
     1 T40,'PRINT OPTION = 'I2///)                                        00205
C * * *  E N T E R I N G   F L O W C H A R T   B L O C K   F   * * * * *  00206
      IF(IDDIF.GT.0) CALL LETTO(&104)                                     00207
C * * *  E N T E R I N G   F L O W C H A R T   B L O C K   G   * * * * *  00208
      CALL VAL(X,DUAL)                                                    00209
C * * *  E N T E R I N G   F L O W C H A R T   B L O C K   H   * * * * *
```

```
      WRITE(6,60) DUAL                                                  00210
   60 FORMAT(T40,60('-')/'1'T40,'DUAL SOLUTION.'/T40,14('=')////        00211
     1T40,'THE VALUE OF THE DUAL FUNCTION AT A MAXIMUM POINT = '/T80,G16 00212
     1T40,'THE CORRESPONDING DUAL VARIABLES ARE0'//T40,60('-')/         00213
     1.8///T40,'THE CORRESPONDING DUAL VARIABLES ARE0'//T40,60('-')/    00214
     1T60,'DUAL',      T78,'VALUE'/T58,'VARIABLE'/T40,60('-'))          00215
      J = CAPT+CAPM                                                     00216
      WRITE(6,179)(I,W(I),I=1,J)                                        00217
  179 FORMAT(T60,'W(',I2,') = ',T70,G16.8)                              00218
C.......                                                                00219
C....... BEGIN CONSTRUCTION OF MATRIX OF EXPONENTS OF PRIMAL VARIABLES. 00220
C.......                                                                00221
C * * *  E N T E R I N G   F L O W C H A R T   B L O C K   I   * * * * * 00222
      IT = TSUBM(1)                                                     00223
      DO 50 T=1,IT                                                      00224
      ROW(T) = T                                                        00225
      COL(T) = T                                                        00226
C.......                                                                00227
C....... A DUAL VARIABLE CLOSE TO ZERO CAUSES EXCESSIVE ROUND-OFF ERROR. 00228
C....... THAT ROW OF THE MATRIX IS SET TO ZERO, AND WILL BE DISCARDED   00229
C....... BY MAT.                                                        00230
C.......                                                                00231
      IF(W(T).LT.10.**(-30)) GO TO 200                                  00232
C.......                                                                00233
C....... IF SIG(1)*DUAL IS NEGATIVE, LET SIG(1) = -SIG(1) AND GO BACK   00234
C....... TO THE DUAL SPACE.  HOWEVER, IF SIG(1)*DJAL = 0, DISREGARD.    00235
C.......                                                                00236
      IF(SIG(1)*DUAL)  193,200,230                                      00237
  230 DO 51 N = 1,CAPN                                                  00238
      B(T,N) = A(1,T,N)                                                 00239
      ...
```

```
C....... THE EQUATIONS FOR SOLUTION OF THE PRIMAL VARIABLES                00240
C....... ARE LINEAR IN LN(X)                                              00241
C.......                                                                  00242
      B(T,CAPNP1) = DLOG(SIG(1)*DUAL*W(T)/XCON(1,T))                      00243
      GO TO 49                                                            00244
  200 DO 201 N = 1,CAPNP1                                                 00245
  201 B(T,N) = 0.0                                                        00246
   49 CONTINUE                                                            00247
   50 CONTINUE                                                            00248
      IF(CAPM.EQ.1) GO TO 54                                              00249
      DO 52 M=2,CAPM                                                      00250
      I=TSUBM(M)                                                          00251
      DO 52 T = 1,I                                                       00252
      IT = IT+1                                                           00253
      ROW(IT) = IT                                                        00254
      COL(IT) = IT                                                        00255
      ATEMP = W(CAPT+M)*XCON(M,T)                                         00256
      IF(DABS(ATEMP).LT.10.**(-30)) GO TO 202                             00257
      ATEMP = W(IT)/ATEMP                                                 00258
      IF(ATEMP.LT.10.**(-30)) GO TO 202                                   00259
      B(IT,CAPNP1) = DLOG(ATEMP)                                          00260
      DO 53 N = 1,CAPN                                                    00261
   53 B(IT,N) =A(M,T,N)                                                   00262
      GO TO 59                                                            00263
  202 DO 203 N = 1,CAPNP1                                                 00264
  203 B(IT,N) = 0.0                                                       00265
   59 CONTINUE                                                            00266
   52 CONTINUE                                                            00267
C.......                                                                  00268
C....... MATRIX OF COEFFICIENTS OF LOGS OF PRIMAL VARIABLES IS COMPLETE.  00269
```

```
C....... CALL MAT TO SOLVE SIMULTANEOUS EQUATIONS FOR PRIMAL VARIABLES.
C.......
C * * *  E N T E R I N G   F L O W C H A R T   B L O C K   J  * * * * *
   54 CALL MAT(CAPT,CAPNP1,&104)
      DO 80 I = 1,CAPN
      IF(B(ROW(I),COL(CAPNP1)).GT.174.673) B(ROW(I),COL(CAPNP1))=174.673
   80 B(ROW(I),COL(CAPNP1)) =DEXP(B(ROW(I),COL(CAPNP1)))
      WRITE (6,74)
C * * *  E N T E R I N G   F L O W C H A R T   B L O C K   K  * * * * *
   74 FORMAT(T40,60('-')////T40,'PRIMAL SOLUTION.'/T40,16('=')////
     1T40,'THE VALUES OF THE PRIMAL VARIABLES AT A MINIMUM POINT ARE ' /
     1T40,'THE VALUES OF THE PRIMAL VARIABLES AT A MINIMUM POINT'/T40,60('-'))
      WRITE (6,73)  (I,B(ROW(I),COL(CAPNP1)) ,I=1,CAPN)
   73 FORMAT(T60,'X(',I2,') = ',T70,G16.8)
      WRITE (6,77)
   77 FORMAT (  T40,60('-')////T40  ,
     1'THE CONTRIBUTION OF EACH TERM TO THE PRIMAL FUNCTION ISO'    //
     1 T40,60('-')/T60,'TERM'T75,'CONTRIBUTION'/T40,60('-'))
C.......
C....... SUBSTITUTE PRIMAL VARIABLES INTO GENERALIZED POLYNOMIAL.
C.......
      ITT = 0
      DO 71 M = 1,CAPM
      Y = 0.0
      IT = TSUBM(M)
      DO 79 T=1,IT
      ITT = ITT + 1
      YTERM = 1.0
      DO 70 N = 1,CAPN
C.......
```

```
C........ IF COEFFICIENT A = 0.0,  X**A = 1.0, UNLESS X.LT.0,                    00300
C........ IN WHICH CASE THE TERM BLOWS UP.                                       00301
C........                                                                        00302
C........                                                                        00303
      IF(A(M,T,N)) 260,70,170                                                    00304
  260 IF(B(ROW(N),COL(CAPNP1)))170,261,170                                       00305
  261 YTERM = YTERM*10.**(30)                                                    00306
      GO TO 70                                                                   00307
  170 YTERM = YTERM*B(ROW(N),COL(CAPNP1))**A(M,T,N)                              00308
   70 CONTINUE                                                                   00309
      YTERM = SIGMA(M,T)*YTERM*XCON(M,T)                                         00310
      WRITE (6,178) ITT,YTERM                                                    00311
  178 FORMAT(T62,I2,T70,G16.8)                                                   00312
   79 Y = Y+YTERM                                                                00313
C........                                                                        00314
C........ YY = VALUE OF THE PRIMAL OBJECTIVE FUNCTION.                          00315
C.........                                                                       00316
      IF(M.EQ.1) YY = Y                                                          00317
   71 WRITE (6,76) M,Y                                                           00318
   76 FORMAT(/T40,'     THE VALUE OF ROW 'I2,' = 'G16.8, /)                      00319
      WRITE (6,75) YY                                                            00320
   75 FORMAT(T4),60('-')/////T40,'THE VALUE OF THE PRIMAL FUNCTION = '            00321
     1 G16.8/)                                                                   00322
C.........                                                                       00323
C........ COMPUTE ABSOLUTE AND RELATIVE ERROR BETWEEN PRIMAL AND DUAL           00324
C........ FUNCTIONS.  IN THE CONVEX CASE, THE SOLUTION IS GUARANTEED            00325
C........ TO LIE BETWEEN THESE FUNCTIONS.                                       00326
C.........                                                                       00327
      D =DABS(YY-DUAL)                                                           00328
      E = (YY+ DUAL)/2.0                                                         00329
      F = D/E
```

```
      WRITE(6,78) DUAL,D,D,E,F                                          00330
   78 FORMAT(T40,'THE VALUE OF THE DUAL FUNCTION = 'G16.8//             00331
     1T40,'THE ABSOLUTE VALUE OF THE DIFFERENCE = 'G16.8//              00332
     1T40,'THE RELATIVE DIFFERENCE = 'G13.6,'/'G13.6//T64,'= '2PF10.6,   00333
     1' PER CENT'/)                                                     00334
      DO 310 M = 1,CAPM                                                 00335
      IF(SIG(M)) 315,311,311                                            00336
  311 IT = TSUBM(M)                                                     00337
      DO 310 T = 1,IT                                                   00338
      IF(SIGMA(M,T)) 315,310,310                                        00339
  310 CONTINUE                                                          00340
      WRITE (6,313)                                                     00341
      GO TO 104                                                         00342
  315 WRITE(6,312)                                                      00343
  313 FORMAT(/T40,'PRIMAL PROBLEM IS CONVEX.  GLOBAL MINIMUM GUARANTEED. 00344
     1'/)                                                               00345
  312 FORMAT(/T40,'PRIMAL PROBLEM IS NON-CONVEX.'                       00346
     1 /T40,'GLOBAL MINIMUM IS NOT GUARANTEED.'/T40,                    00347
     1 'THIS STATIONARY POINT MAY BE A LOCAL MINIMUM, SADDLE POINT, OR M 00348
     1AXIMUM.'/)                                                        00349
C........                                                               00350
C....... TO TERMINATE, PUT 1 IN COL 2 AND LEAVE REST BLANK.             00351
C....... CAPM MUST NOT = 0 OR ERROR WILL RESULT.                        00352
C....... IF CAPN = 0, PROGRAM WILL TERMINATE.                           00353
C....... OTHERWISE, IT WILL GO BACK TO 92.                              00354
C........                                                               00355
  104 READ (5,1) CAPM,CAPN,(TSUBM(M),M=1,CAPM),(SIG(M),M=1,CAPM)        00356
C........                                                               00357
C....... CLOCK IS NOT AVAILABLE IN WATFOR                               00358
C........                                                               00359
```

```
      CALL CLOCK(HRS,AMIN,BSEC)                                          00360
      ATIME = BSEC - ASEC                                                00361
      WRITE(6,111) ATIME                                                 00362
  111 FORMAT(T40,'ELAPSED TIME = ',F6.3,' SECONDS.'/'1')                 00363
      IF(CAPN) 101,101,250                                               00364
C.....                                                                   00365
C..... RE-INITIALIZE CERTAIN MATRICES TO 0.                              00366
C..... CLEAR AND CLEARD ARE NOT AVAILABLE IN WATFOR                      00367
C.....                                                                   00368
C.....                                                                   00369
  250 CALL CLEARD(A(1,1,1),1000)                                         00370
      CALL CLEARD(XCON(1,1),100)                                         00371
      CALL CLEAR(SIGMA(1,1),100)                                         00372
      DO 91 I = 1,32                                                     00373
      ROW(I) = I                                                         00374
      COL(I) = I                                                         00375
   91 CONTINUE                                                           00376
      GO TO 92                                                           00377
  101 STOP                                                               00378
      END                                                                00379
      SUBROUTINE MAT(NROWS,NCOLS,*)                                      00380
C.....                                                                   00381
C..... THIS SUBROUTINE PLACES AN IDENTITY MATRIX IN THE LEFT-HAND        00382
C..... SQUARE MATRIX OF B , WHICH MAY BE SQUARE OR RECTANGULAR.          00383
C.....                                                                   00384
      IMPLICIT REAL*8(A-B,D-H,O-Q,U-Z),  INTEGER(C,R-T)                  00385
      COMMON/TOM/A(10,10,10),B(31,32),XCON(10,10),SIGMA(10,10),SIG(10),  00386
     1ROW(32),COL(32),TSUBM(10),CAPM,CAPN,CAPNP1,CAPNP2,CAPT,CAPTP1      00387
      NCM1 = NCOLS - 1                                                   00388
C.....                                                                   00389
C..... CREATE UPPER-TRIANGULAR MATRIX.
```

```
                                                                      00390
C.........                                                            00391
C * * *  E N T E R I N G   F L O W C H A R T   B L O C K   L  * * * * *  00392
      DO 10 IPIV = 1,NROWS                                            00393
C.........                                                            00394
C........ DIVIDE ROW BY PIVOT ELEMENT.                                00395
C.........                                                            00396
      IPP1 = IPIV+1                                                   00397
      IF(B(ROW(IPIV),COL(IPIV)).EQ.1.0) GO TO 23                      00398
      DO 31 TT=IPIV,NCM1                                              00399
      DO 11 T = IPIV,NROWS                                            00400
C.........                                                            00401
C........ IF PIVOT ELEMENT = 0.0, EXCHANGE ROWS.                      00402
C.........                                                            00403
C * * *  E N T E R I N G   F L O W C H A R T   B L O C K   M  * * * * *  00404
      IF(B(ROW(IPIV),COL(IPIV))) 12,40,12                             00405
C.........                                                            00406
C........ IF ALL ROWS HAVE BEEN EXCHANGED, TRY EXCHANGING COLUMNS.    00407
C.........                                                            00408
C * * *  E N T E R I N G   F L O W C H A R T   B L O C K   N  * * * * *  00409
   40 IF(T.EQ.NROWS)  GO TO 30                                        00410
C * * *  E N T E R I N G   F L O W C H A R T   B L O C K   O  * * * * *  00411
      IT = ROW(IPIV)                                                  00412
      ROW(IPIV) = ROW(T+1)                                            00413
   11 ROW(T+1) = IT                                                   00414
C.........                                                            00415
C........ IF THERE ARE NO NON-ZERO ELEMENTS LEFT, TERMINATE.          00416
C.........                                                            00417
C * * *  E N T E R I N G   F L O W C H A R T   B L O C K   P  * * * * *  00418
   30 IF(TT.EQ.NCM1) GO TO 104                                        00419
C * * *  E N T E R I N G   F L O W C H A R T   B L O C K   Q  * * * * *
```

```
      IT = COL(IPIV)                                                    00420
      COL(IPIV) = COL(TT+1)                                             00421
      COL(TT+1) = IT                                                    00422
   31 COL(TT+1) = IT                                                    00423
C * * *  E N T E R I N G   F L O W C H A R T   B L O C K   R   * * * * *  00424
   12 DO 13 T = IPP1, NCOLS                                             00425
   13 B(ROW(IPIV),COL(T))=B(ROW(IPIV),COL(T))/B(ROW(IPIV),COL(IPIV))    00426
      B(ROW(IPIV),COL(IPIV)) = 1.0                                      00427
C.........                                                            00428
C........ THE ROW HAS BEEN DIVIDED BY THE PIVOT ELEMENT.              00429
C........ IF THIS COMPLETES THE UPPER-TRIANGULAR MATRIX , GO TO 14.   00430
C.........                                                            00431
   23 IF(IPIV.EQ.NROWS.OR.IPIV.EQ.NCM1) GO TO 14                        00432
C.........                                                            00433
C........ PLACE ZEROS IN THE ELEMENTS BELOW THE PIVOT ELEMENT.        00434
C.........                                                            00435
      DO 21 N = IPP1,NROWS                                              00436
C.........                                                            00437
C........ IF THE ELEMENT IS ALREADY ZERO, SKIP THAT ROW.              00438
C.........                                                            00439
      IF(B(ROW(N),COL(IPIV))) 41,21,41                                  00440
C * * *  E N T E R I N G   F L O W C H A R T   B L O C K   S   * * * * *  00441
   41 IT = NCOLS+1-IPIV                                                 00442
      DO 15 I=1,IT                                                      00443
      T = NCOLS+1-I                                                     00444
   15 B(ROW(N),COL(T))=B(ROW(N),COL(T))/B(ROW(N),COL(IPIV))             00445
     1 -B(ROW(IPIV),COL(T))                                             00446
   21 CONTINUE                                                          00447
C * * * * *  T H I S   I S   F L O W C H A R T   P O I N T   1 0   * * * * *  00448
   10 CONTINUE                                                          00449
C.........
```

```
C....... UPPER-TRIANGULAR MATRIX COMPLETED.                              00450
C....... IF THERE IS ONLY ONE ROW OR ONE COLUMN, RETURN.                 00451
C.....                                                                   00452
C.....                                                                   00453
   14 IF(NROWS.EQ.1.OR.NCM1.EQ.1) RETURN                                 00454
C.....                                                                   00455
C....... IF THERE ARE MORE ROWS THAN COLUMNS, DISCARD THE BOTTOM ROWS.   00456
C.....                                                                   00457
      IF(NROWS.GT.NCM1) NROWS = NCM1                                     00458
      NRP1 = NROWS + 1                                                   00459
C.....                                                                   00460
C....... PLACE ZEROS IN UPPER ELEMENTS.                                  00461
C....... PIVOT ELEMENTS START AT THE BOTTOM.                             00462
C.....                                                                   00463
C * * *  E N T E R I N G   F L O W C H A R T   B L O C K   T  * * * * *  00464
      DO 20 T = 2,NROWS                                                  00465
      IPIV = NROWS+2-T                                                   00466
      IPM1 = IPIV-1                                                      00467
      DO 20 N=1,IPM1                                                     00468
C.....                                                                   00469
C....... IF THERE IS ALREADY A ZERO IN THIS ELEMENT, SKIP THIS ROW.      00470
C.....                                                                   00471
      IF(B(ROW(N),COL(IPIV))) 42,20,42                                   00472
C * * *  E N T E R I N G   F L O W C H A R T   B L O C K   U  * * * * *  00473
   42 BCOEF = B(ROW(N),COL(IPIV))/B(ROW(IPIV),COL(IPIV))                 00474
      DO 22 TCOL=NRP1,NCOLS                                              00475
      B(ROW(N),COL(TCOL)) = B(ROW(N),COL(TCOL))                          00476
     1 - B(ROW(IPIV),COL(TCOL))*BCOEF                                    00477
   22 CONTINUE                                                           00478
      B(ROW(N),COL(IPIV)) = 0.0                                          00479
   20 CONTINUE
```

```
C.........                                                              00480
C........ IDENTITY MATRIX COMPLETED.                                   00481
C.........                                                              00482
C.........                                                              00483
      RETURN                                                           00484
  104 WRITE (6,45)                                                     00485
   45 FORMAT(//T40,'SINGULARITY DISCOVERED IN MATRIX***************'//)00486
      DO 44 I = 1,NROWS                                                00487
   44 WRITE (6,43) (B(ROW(I),COL(J)),J = 1,NCOLS)                      00488
   43 FORMAT(T43,13F7.3/T43,9F7.3)                                     00489
      WRITE(6,46)                                                      00490
   46 FORMAT(//T40,'THIS IS A PATHOLOGICAL PROBLEM.'//                 00491
     1 T40,'IT IS SUGGESTED THAT THE PRIMAL PROBLEM EITHER HAS NO'     00492
     1 /T40,'MINIMUM OR HAS AN OBVIOUS AND ''TRIVIAL'' MINIMUM.'/      00493
     1 T40,'EXECUTION TERMINATED*************************************'/)00494
      RETURN 1                                                         00495
      END                                                              00496
      SUBROUTINE BLOWUP(I,*)                                           00497
C.........                                                             00498
C........ THIS SUBROUTINE IS CALLED FROM THE MAIN PROGRAM AND WILL     00499
C......... TERMINATE PROBLEM EXECUTION.                                00500
C.........                                                             00501
      IMPLICIT REAL*8(A-B,D-H,O-Q,U-Z),  INTEGER(C,R-T)                00502
      GO TO (1,2,3),I                                                  00503
    1 WRITE(6,10)                                                      00504
      GO TO 100                                                        00505
    2 WRITE(6,20)                                                      00506
      GO TO 100                                                        00507
    3 WRITE(6,30)                                                      00508
   10 FORMAT(/T40,'ABOVE DATA IMPLIES DATA CARD ERROR.'/)               00509
   20 FORMAT(/T40,'ABOVE DATA EXCEEDS UPPER LIMITS ON SIZE OF PROBLEM')
```

```
      D = 0.01                                                          00540
      ADAF = 1.0                                                        00541
      CFSC = 3.1314159                                                  00542
      CASS = CFSC                                                       00543
      ISTOP = 2                                                         00544
      IFEAS = 1                                                         00545
      KC = 0                                                            00546
      LACT = 0                                                          00547
      CALL SUMS(X, ZBASE)                                               00548
      IF(LPRINT.LE.3)  WRITE (6, 1110)                                  00549
 1110 FORMAT( T40, 'INITIAL POINT.')                                    00550
      IF(LPRINT.LE.3)  CALL OUTPUT (X, ZBASE)                           00551
      ZTEMP = ZBASE                                                     00552
      DO 110 I = 1, N                                                   00553
      XTEMP(I) = X(I)                                                   00554
  110 XBASE(I) = X(I)                                                   00555
      IF (IFEAS .EQ. 1) GO TO 111                                       00556
      IF(LPRINT.LE.2)   WRITE (6, 5110)                                 00557
 5110 FORMAT(T40,'INITIAL POINT NOT IN FEASIBLE REGION.')               00558
      CALL SUMC(X, CBASE)                                               00559
      CTEMP = CBASE                                                     00560
      CFSCI = CFSC                                                      00561
C**** THIS IS POINT A WHICH APPEARS IN FLOWCHART, FIG. 8.               00562
 5111 DO 5112 I = 1,N                                                   00563
 5112 XTEST(I) = XBASE(I)                                               00564
 5114 DO 5115 I = 1, N                                                  00565
      XTEST(I) = XTEMP(I) + CFSCI                                       00566
      CALL SUMC (XTEST, CTEST)                                          00567
      IF (CTEST .GT. CTEMP) GO TO 5135                                  00568
      XTEST(I) = XTEMP(I) - CFSCI                                       00569
```

```
 5174 XBASE(I) = XTEST(I)                                                00600
      GO TO 5175                                                         00601
C**** THIS IS POINT B WHICH APPEARS IN FLOWCHART, FIG. 10.               00602
 5146 DO 5147 I = 1,N                                                    00603
      X(I) = XTEMP(I)                                                    00604
 5147 XBASE(I) = XTEMP(I)                                                00605
C**** THIS IS POINT F WHICH APPEARS IN FLOWCHART, FIG. 10.               00606
 5176 IF(LPRINT.LE.2)   WRITE (6,5177)                                   00607
 5177 FORMAT(T40,'FEASIBLE POINT HAS BEEN FOUND.')                       00608
      CALL VAL (X, ZBASE)                                                00609
      ZTEMP = ZBASE                                                      00610
      GO TO 111                                                          00611
 5210 IF(LPRINT.LE.1)   WRITE (6,5211)                                   00612
 5211 FORMAT(T4),'MORE FEASIBLE POINT NOT FOUND IN LOCAL SEARCH.'/       00613
     1T40,'REDUCE STEP SIZE (CFSCI).')                                   00614
      CTEMP = CBASE                                                      00615
      DO 5216 I = 1,N                                                    00616
 5216 XTEMP(I) = XBASE(I)                                                00617
      CFSCI = CFSCI/3.0                                                  00618
      IF(CFSCI .GE. XLLIM) GO TO 5111                                    00619
      IF(LPRINT.LE.3)  WRITE(6,5270)                                     00620
 5270 FORMAT(T4), 'NO FEASIBLE POINT FOUND. PROBLEM TERMINATED.')        00621
      RETURN 1                                                           00622
C**** THIS IS POINT D WHICH APPEARS IN FLOWCHART, FIG. 10.               00623
 5172 IF(LPRINT.LE.1)WRITE(6,6172)                                       00624
 6172 FORMAT(T40,'BETTER POINT NOT FOUND IN PATTERN MOVE.'/              00625
     1T40,'RETURN TO BEST POINT IN LOCAL SEARCH.')                       00626
      CBASE = CTEMP                                                      00627
      DO 5195 I = 1,N                                                    00628
      X(I) = XTEMP(I)                                                    00629
```

```
 5195 XBASE(I) = XTEMP(I)                                              00630
      GO TO 5111                                                       00631
C**** THIS IS POINT 111 WHICH APPEARS IN FLOWCHART, FIG. 11.           00632
  111 DO 112 I = 1, N                                                  00633
      XTEMP(I) = XBASE(I)                                              00634
  112 XTEST(I) = XBASE(I)                                              00635
      ZTEMP = ZBASE                                                    00636
  114 DO 115 I = 1, N                                                  00637
      XTEST(I) = XTEMP(I) + CFSC                                       00638
      CALL SUMS(XTEST, ZTEST)                                          00639
      IF (ZTEST .GT. ZTEMP) GO TO 135                                  00640
      XTEST(I) = XTEMP(I) - CFSC                                       00641
      CALL SUMS (XTEST, ZTEST)                                         00642
      IF (ZTEST .LE. ZTEMP) GO TO 115                                  00643
  135 ZTEMP = ZTEST                                                    00644
      XTEMP(I) = XTEST(I)                                              00645
  115 CONTINUE                                                         00646
C**** THIS IS POINT G WHICH APPEARS IN FLOWCHART, FIG. 12.             00647
      CALL SUMS(XTEMP, ZTEST)                                          00648
      IF(LPRINT.LE.0)   CALL OUTPUT (XTEMP, ZTEST)                     00649
  145 IF (IFEAS .EQ. 2) GO TO 146                                      00650
  150 IF (ZTEST .LE. ZBASE + XLLIM) GO TO 149                          00651
      ZTEMP = ZTEST                                                    00652
      CASS = CFSC                                                      00653
      ISTOP = 2                                                        00654
      ADAF = 1                                                         00655
      DO 160 I = 1,N                                                   00656
  160 XTEST(I) = XTEMP(I)*2. -X(I)                                     00657
      CALL SUMS (XTEST, ZTEST)                                         00658
      IF(LPRINT.LE.0)   CALL OUTPUT (XTEST,ZTEST)                      00659
```

```
  170 IF (ZTEST.LE. ZTEMP+ XLLIM) GO TO 172                                      00660
  171 IF (IFEAS .EQ. 2) GO TO 173                                                00661
      ZBASE = ZTEST                                                              00662
      ZTEMP = ZTEST                                                              00663
      ADAF = 1.0                                                                 00664
      DO 180 I = 1,N                                                             00665
      X(I) = XTEMP(I)                                                            00666
  180 XBASE(I) = XTEST(I)                                                        00667
      IF(LPRINT.LE.1)   WRITE (6,1180)                                           00668
 1180 FORMAT(T40, 'LET PATTERN MOVE STAND.')                                     00669
      GO TO 111                                                                  00670
  173 IF(LPRINT.LE.1)   WRITE (6, 1173)                                          00671
 1173 FORMAT(T40,'CONSTRAINT CONTACTED IN PATTERN MOVE.',    /                   00672
     1T40,'RETURN TO BEST POINT IN LOCAL SEARCH.')                               00673
      GO TO 190                                                                  00674
C**** THIS IS POINT K WHICH APPEARS IN FLOWCHART, FIG. 13.                       00675
  172 IF(LPRINT.LE.1)   WRITE (6, 1172)                                          00676
 1172 FORMAT(T40,'BETTER POINT NOT FOUND IN PATTERN MOVE.'/                      00677
     1T40,'RETURN TO BEST POINT IN LOCAL SEARCH.')                               00678
C**** THIS IS POINT L WHICH APPEARS IN FLOWCHART, FIG. 13.                       00679
                                                                                 00680
  190 ZBASE = ZTEMP                                                              00681
      DO 195 I = 1, N                                                            00682
      X(I) = XTEMP(I)                                                            00683
  195 XBASE(I) = XTEMP(I)                                                        00684
      GO TO 111                                                                  00685
  151 IF(LPRINT.LE.1)   WRITE (6, 1151)                                          00686
 1151 FORMAT(T40,'MULTIPLE CONSTRAINTS CONTACTED IN LOCAL SEARCH.'/T40,          00687
     1'REDUCE STEP SIZE (CFSC) TO HAVE ONE CONSTRAINT IN CONSTRAINED SEA         00688
     1RCH.')                                                                     00689
      GO TO 220
```

```
C**** THIS IS POINT H WHICH APPEARS IN FLOWCHART, FIG. 13.                    00690
  146 ZTEMP = ZBASE                                                          00691
      DO 147 I = 1,N                                                         00692
  147 XTEMP(I) = XBASE(I)                                                    00693
      IF (KC .NE. 1) GO TO 151                                               00694
  148 IF(LPRINT.LE.2)   WRITE (6, 1148)                                      00695
 1148 FORMAT(T40,'SINGLE CONSTRAINT CONTACTED IN LOCAL SEARCH.')             00696
C**** THIS IS POINT 210 WHICH APPEARS IN FLOWCHART, FIG. 14.                 00697
  210 IF (ISTOP .EQ. 2) GO TO 409                                            00698
C**** THIS IS POINT M WHICH APPEARS IN FLOWCHART, FIG. 14.                   00699
  220 CFSC = CFSC/3.0                                                        00700
      IF (CFSC .GE. XLLIM) GO TO 231                                         00701
      IF(LPRINT.LE.3)   WRITE (6, 2232)                                      00702
 2232 FORMAT(T40,'INCREMENTAL VALUE OF VARIABLES .LT. LOWER LIMIT.',         00703
     1/T40,'PATTERN SEARCH TERMINATED.')                                     00704
      CALL SUMS (XTEMP, ZTEMP)                                               00705
      IF(LPRINT.LE.3)   CALL OUTPUT (XTEMP, ZTEMP)                           00706
      GO TO 9999                                                             00707
C**** THIS IS POINT J WHICH APPEARS IN FLOWCHART, FIG. 13.                   00708
  149 IF(LPRINT.LE.1)   WRITE (6, 1149)                                      00709
 1149 FORMAT(T40, 'BETTER POINT NOT FOUND IN LOCAL SEARCH.'/                 00710
     1T40,'REDUCE STEP SIZE (CFSC).')                                        00711
C**** THIS IS POINT N WHICH APPEARS IN FLOWCHART, FIG. 13.                   00712
  215 ZTEMP = ZBASE                                                          00713
      DO 216 I = 1,N                                                         00714
  216 XTEMP(I) = XBASE(I)                                                    00715
      GO TO 220                                                              00716
  231 IF(LPRINT.LE.1)   WRITE (6, 2231)                                      00717
 2231 FORMAT(T40,'START LOCAL SEARCH WITH REDUCED STEP SIZE (CFSC).')        00718
      GO TO 111                                                              00719
```

```
C**** THIS IS POINT 310 WHICH APPEARS IN FLOWCHART, FIG. 17.                    00720
  310 CONTINUE                                                                  00721
  329 DO 330 I = 1, N                                                           00722
  330 XTEST(I) = XBASE(I) + ADAF*CASS*PARTH(I)                                  00723
      CALL SUMS (XTEST, ZTEST)                                                  00724
      IF(LPRINT.LE.0)   CALL OUTPUT (XTEST, ZTEST)                              00725
  350 GO TO (355, 380),IFEAS                                                    00726
  355 IF(LPRINT.LE.1)   WRITE (6, 3355)                                         00727
 3355 FORMAT(T40 ,'NO CONSTRAINT CONTACTED IN CONSTRAINED MOVE.')               00728
  360 IF (ZTEST .GT. ZBASE) GO TO 370                                           00729
      IF (ADAF .LE. 1.1) GO TO 362                                              00730
      IF(LPRINT.LE.1)   WRITE (6, 3365)                                         00731
 3365 FORMAT(T40,'ACCELERATED CONSTRAINED MOVE NOT A BETTER POINT.'/            00732
     1T40,'LET CONSTRAINED SEARCH ACCELERATION FACTOR(ADAF) = 1.0')             00733
      ADAF = 1.0                                                                00734
      GO TO 329                                                                 00735
  362 IF(LPRINT.LE.1)   WRITE (6, 3362)                                         00736
 3362 FORMAT(T40,'CONSTRAINT VIOLATED IN CONSTRAINED MOVE.'                     00737
     1 '  REDUCE STEP SIZE (CASS).')                                           00738
      GO TO 381                                                                 00739
  380 IF(LPRINT.LE.1)   WRITE (6, 3380)                                         00740
 3380 FORMAT(T40,'CONSTRAINT CONTACTED IN CONSTRAINED SEARCH.  REDUCE ST        00741
     1EP SIZE (CASS).'  )                                                       00742
      ADAF = 1.0                                                                00743
  381 CASS = CASS/3.0                                                           00744
  385 IF (CASS .GT. XLLIM) GO TO 310                                            00745
      IF(LPRINT.LE.2)   WRITE (6, 3385)                                         00746
 3385 FORMAT(T40,'CASS .LT. MIN DELTA X. CONSTRAINED SEARCH HAS FAILED.'        00747
     1)                                                                         00748
  390 ISTOP = 1                                                                 00749
```

```
      GO TO 215                                                          00750
  370 ADAF = ADAF + 1.0                                                  00751
      ZBASE = ZTEST                                                      00752
      ZTEMP = ZTEST                                                      00753
      DO 375 I = 1, N                                                    00754
      X(I) = XTEST(I)                                                    00755
      XBASE(I) = XTEST(I)                                                00756
  375 XTEMP(I) = XBASE(I)                                                00757
      IF(LPRINT.LE.1)   WRITE (6, 3377)                                  00758
 3377 FORMAT(T4,'CONSTRAINED MOVE SUCCESSFUL. START ANOTHER LOCAL SEARC  00759
     1H.')                                                               00760
      GO TO 111                                                          00761
C**** THIS IS POINT P WHICH APPEARS IN FLOWCHART, FIG. 15.               00762
  409 DO 410 I = 1, N                                                    00763
  410 X(I) = XBASE(I)                                                    00764
      CASS = CFSC                                                        00765
      DO 426 I = 1,N                                                     00766
      X(I) = X(I) + D                                                    00767
      CALL VAL(X, ZP)                                                    00768
      CP = C(LACT)                                                       00769
      X(I) = X(I) - 2.0 * D                                              00770
      CALL VAL(X, ZM)                                                    00771
      CM = C(LACT)                                                       00772
  422 PARTZ(I) = (ZP-ZM)                                                 00773
      PARTH(I) = (CP-CM)                                                 00774
      X(I) = XBASE(I)                                                    00775
  426 CONTINUE                                                           00776
      DENOM = 0.0                                                        00777
      DO 430 I = 1, N                                                    00778
  430 DENOM = DENOM + PARTZ(I)**2                                        00779
```

```
      DENOM = DSQRT(DENOM)                                                      00780
      IF (DENOM .LT. XLLIM) DENOM = XLLIM                                       00781
      DO 440 I = 1,N                                                            00782
      PARTZ(I) = PARTZ(I)/DENOM                                                 00783
  440 CONTINUE                                                                  00784
      IF(LPRINT.LE.2.AND.N.LE.6) WRITE(6,4441) (PARTZ(I),I=1,N)                 00785
 4441 FORMAT(T40,'OBJECTIVE GRADIENT IS '  6F10.6)                              00786
      IF(LPRINT.LE.2.AND.N.GT.6) WRITE(6,9441) (PARTZ(I),I=1,N)                 00787
 9441 FORMAT(T20,'OBJECTIVE GRADIENT= '9F10.6/T20,11F10.6)                      00788
C**** THIS IS POINT Q WHICH APPEARS IN FLOWCHART, FIG. 16.                      00789
      DENOM = 0.0                                                               00790
      DO 460 I = 1, N                                                           00791
  460 DENOM = DENOM + PARTH(I)**2                                               00792
      DENOM = DSQRT(DENOM)                                                      00793
      DO 470 I = 1, N                                                           00794
  470 PARTH(I) = PARTH(I)/DENOM                                                 00795
      IF(LPRINT.LE.2.AND.N.LE.6) WRITE (6, 4475) (PARTH(I), I = 1,N)            00796
 4475 FORMAT(T40,'CONSTRAINT GRADIENT IS' 6F10.6)                               00797
      IF(LPRINT.LE.2.AND.N.GT.6) WRITE(6,9442) (PARTH(I),I=1,N)                 00798
 9442 FORMAT(T20,'CONSTRAINT GRADIENT=' 9F10.6/T20,11F10.6)                     00799
      DENOM = 0.0                                                               00800
      DO 480 I = 1, N                                                           00801
      PARTH(I) = PARTH(I) + PARTZ(I)                                            00802
  480 DENOM = DENOM + PARTH(I)**2                                               00803
      DENOM = DSQRT(DENOM)                                                      00804
      IF (DENOM .LT. XLLIM) DENOM = XLLIM                                       00805
      DO 490 I = 1, N                                                           00806
  490 PARTH(I) = PARTH(I)/DENOM                                                 00807
      IF(LPRINT.LE.2.AND.N.LE.6)    WRITE (6, 4495) (PARTH(I), I = 1, N)        00808
 4495 FORMAT(T40,'COMBINED GRADIENT IS  ' 6F10.6)                               00809
```

```
      IF(LPRINT.LE.2.AND.N.GT.6) WRITE(6,9495)(PARTH(I),I=1,N)                00810
      IF(LPRINT.LE.2.AND.N.GT.6) WRITE(6,9495)(PARTH(I),I=1,N)                00811
 9495 FORMAT(T20,'COMBINED GRADIENT = ' 9F10.6/T20,11F10.6)                   00812
      IF (DENOM .GT. 0.03) GO TO 310                                          00813
      IF(LPRINT.LE.2)   WRITE (6, 4498)                                       00814
 4498 FORMAT (T40,'OBJECTIVE AND CONSTRAINT NEARLY PARALLEL.')                00815
      ISTOP = 1                                                               00816
      GO TO 111                                                               00817
 9999 CONTINUE                                                                00818
      RETURN                                                                  00819
      END                                                                     00820
      SUBROUTINE SUMS(X, Z)                                                   00821
C**** THIS SUBROUTINE APPEARS IN FLOWCHART, FIG. 4.                           00822
      IMPLICIT REAL*8(A-H,O-Z)                                                00823
      DIMENSION X(20)                                                         00824
      COMMON C(41),CFSC,CASS,ADAF,XLLIM,KC,IFEAS,ISTOP,LACT,M,N,LPRINT        00825
      CALL VAL(X, Z)                                                          00826
      KC = 0                                                                  00827
      IFEAS = 1                                                               00828
      IF (M .LT. 1) GO TO 11                                                  00829
      DO 10 I = 1, M                                                          00830
      IF (C(I) .GE. 0.0) GO TO 10                                             00831
   40 IFEAS = 2                                                               00832
      LACT = I                                                                00833
   50 KC = KC + 1                                                             00834
   10 CONTINUE                                                                00835
   11 CONTINUE                                                                00836
      RETURN                                                                  00837
      END                                                                     00838
      SUBROUTINE SUMC(X, CON)                                                 00839
C**** THIS SUBROUTINE APPEARS IN FLOWCHART, FIG. 5.
```

```
      IMPLICIT REAL*8(A-H,O-Z)                                          00840
      DIMENSION X(20)                                                   00841
      COMMON C(41),CFSC,CASS,ADAF,XLLIM,KC,IFEAS,ISTOP,LACT,M,N,LPRINT  00842
      CALL VAL (X, Z)                                                   00843
      CON = 0.0                                                         00844
      IFEAS = 1                                                         00845
      DO 10 I = 1,M                                                     00846
      IF(C(I) .GE. 0.0) GO TO 9                                         00847
      IFEAS = 2                                                         00848
      CON = CON + C(I)                                                  00849
    9 CONTINUE                                                          00850
   10 CONTINUE                                                          00851
   11 CONTINUE                                                          00852
      RETURN                                                            00853
      END                                                               00854
      SUBROUTINE OUTPUT(X, Z)                                           00855
C                                                                       00856
C..... THIS SJBROUTINE PRINTS OUT VALUES OF THE PATTERN SEARCH          00857
C..... VARIABLES AND CONSTRAINTS AT THE END OF EACH SUCCESSFUL STEP     00858
C..... IF LPRINT = 0.                                                   00859
C                                                                       00860
      IMPLICIT REAL*8(A-H,O-Z)                                          00861
      DIMENSION X(20)                                                   00862
      COMMON C(41),CFSC,CASS,ADAF,XLLIM,KC,IFEAS,ISTOP,LACT,M,N,LPRINT  00863
      IF(N.GT.6) GO TO 40                                               00864
      WRITE (6,100) Z,(X(I),I=1,N)                                      00865
      IF(M.GT.8) GO TO 50                                               00866
      WRITE(6,200) (C(J),J=1,M)                                         00867
      WRITE (6,300)                                                     00868
      RETURN                                                            00869
```

```
   50 WRITE(6,600) (C(J),J = 1,M)                                        00870
      WRITE(6,300)                                                       00871
      RETURN                                                             00872
   40 WRITE(6,400) Z,(X(I),I = 1,N)                                      00873
      WRITE(6,500) (C(J),J = 1,M)                                        00874
      WRITE(6,300)                                                       00875
  100 FORMAT(T40,'OBJ.FCN.=' 1PE12.5,' VARIAB.='0P6F10.6)                00876
  200 FORMAT(T40,'CONSTR.='8F10.6)                                       00877
  300 FORMAT(T40,60('-'))                                                00878
  400 FORMAT(T15,'OBJ.FCN.=' 1PE12.5,' VAR.='9F10.6,/T15,11F10.6)        00879
  500 FORMAT(T15,'CONSTR.='10F10.6/,2(T15,10F10.6/),T15,11F10.6)         00880
  600 FORMAT(T40,'CONSTR.= '8F10.6,/,4(T40,9F10.6/))                     00881
      RETURN                                                             00882
      END                                                                00883
      SUBROUTINE VAL (X,DUAL)                                            00884
C.........                                                              00885
C.......THIS SUBROUTINE CALCULATES THE VALUES OF ALL THE DUAL VARIABLES, 00886
C....... GIVEN THE LAST IDDIF OF THEM, AND THEN CALCULATES THE VALUE     00887
C....... OF THE DUAL FUNCTION.                                          00888
C.........                                                              00889
      IMPLICIT REAL*8(A-B,D-H,O-Q,U-Z),  INTEGER(C,R-T)                  00890
      DIMENSION X(20)                                                    00891
      COMMON/TOM/A(10,10,10),B(31,32),XCON(10,10),SIGMA(10,10),SIG(10),  00892
     1ROW(32),COL(32),TSUBM(10),CAPM,CAPN,CAPNP1,CAPNP2,CAPT,CAPTP1      00893
      COMMON W(41),XCFSC,XCASS,ADAF,XLLIM,KC,IFEAS,ISTOP,LACT,MPT,IDDIF, 00894
     1LPRINT                                                             00895
      IF(IDDIF.EQ.0) X(1) = 0.0                                          00896
      DO 1 I=1,CAPNP1                                                    00897
      BSUM = B(I,CAPTP1)                                                 00898
      DO 2 J=CAPNP2,CAPT                                                 00899
```

```
C.........                                                              00900
C.......THE LAST IDDIF DUAL VARIABLES W = THE PATTERN SEARCH VARIABLES X.  00901
C.........                                                              00902
C.........                                                              00903
      W(J)=X(J-CAPNP1)                                                  00904
    2 BSUM = BSUM - B(I,J)*W(J)                                         00905
C.........                                                              00906
C........ SOLVE FOR THE FIRST DUAL VARIABLES.                           00907
C.........                                                              00908
    1 W(I) = BSUM                                                       00909
      IF(CAPM.EQ.1) GO TO 6                                             00910
      I = TSUBM(1)                                                      00911
C.........                                                              00912
C........ THERE IS ONE ADDITIONAL DUAL CONSTRAINT FOR EACH PRIMAL       00913
C........ CONSTRAINT.                                                   00914
C.........                                                              00915
      DO 3 IM = 2,CAPM                                                  00916
      DSUM = 0.0                                                        00917
      ITS = TSUBM(IM)                                                   00918
      DO 4 IT = 1,ITS                                                   00919
      I=I+1                                                             00920
      DS = SIGMA(IM,IT)                                                 00921
    4 DSUM = DSJM+DS*W(I)                                               00922
      DS = SIG(IM)                                                      00923
    3 W(CAPT+IM) = DSUM*DS                                              00924
    6 W(CAPT+1) = 1.0                                                   00925
C.........                                                              00926
C........ SOLVE FOR THE DUAL FUNCTION.                                  00927
C.........                                                              00928
      I = 0                                                             00929
      DUAL = 1.0
```

```
      DO 5 IM = 1,CAPM                                                  00930
      ITS = TSUBM(IM)                                                   00931
      DO 5 IT = 1,ITS                                                   00932
      I = I+1                                                           00933
      DTERM = 1.0                                                       00934
C.........                                                              00935
C....... IF W(I).= 0.0, THE TERM EQUALS ONE.                            00936
C.........                                                              00937
      IF(W(I)) 10,5,10                                                  00938
   10 Y = XCON(IM,IT)*W(CAPT+IM)/W(I)                                   00939
      IF(Y) 5,11,12                                                     00940
   11 DTERM = 0.0                                                       00941
      IF(SIG(1).LE.0)  DTERM = 10.**(-30)                               00942
      GO TO 5                                                           00943
   12 Z = SIGMA(IM,IT)*W(I)                                             00944
      DTERM = Y**Z                                                      00945
    5 DUAL = DUAL*DTERM                                                 00946
      IF(DUAL) 20,21,22                                                 00947
   20 DUAL = -1.0*(10.**(30))                                           00948
      RETURN                                                            00949
   21 DUAL = 0.0                                                        00950
      IF(SIG(1).LE.0)  DUAL = 10.**(30)                                 00951
      RETURN                                                            00952
   22 DUAL = SIG(1)*(DUAL**SIG(1))                                      00953
      RETURN                                                            00954
      END                                                               00955
      BLOCK DATA                                                        00956
      IMPLICIT REAL*8(A-B,D-H,O-Q,U-Z),  INTEGER(C,R-T)                 00957
      COMMON/TOM/A(10,10,10),B(31,32),XCON(10,10),SIGMA(10,10),SIG(10), 00958
     1ROW(32),COL(32),TSUBM(10),CAPM,CAPN,CAPNP1,CAPNP2,CAPT,CAPTP1     00959
```

```
      COMMON/GUS/X(20),XBASE(20),XTEMP(20),XTEST(20),PARTZ(20),PARTH(20)   00960
      DATA A,XCON,X,SIGMA,ROW,COL/1100*0.0,20*0.2,100*0,                   00961
     1 1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25   00962
     1,26,27,28,29,30,31,32,                                               00963
     1 1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25   00964
     1,26,27,28,29,30,31,32/                                               00965
        END                                                                00966
```

APPENDIX B

EXAMPLE OF OUTPUT

POLYNOMIAL MINIMIZATION BY GEOMETRIC PROGRAMMING

SEA POWER PROBLEM - CUFFIN, ZENER, AND PETERSON - 10**6 WATTS

PRIMAL PROGRAM.

THE NUMBER OF PRIMAL VARIABLES = 7
THE NUMBER OF PRIMAL CONSTRAINTS = 4

*** GENERALIZED POLYNOMIAL ***

ROW	TERM	COEFFICIENT OF TERM	PRIMAL VARIABLE	EXPONENT OF VARIABLE	SIGNUM OF ROW

MINIMIZE.

1	1	1.0000000	* X(1) **	1.0000000	

SIGN OF PRIMAL FUNCTION AT MINIMUM ASSUMED TO BE POSITIVE

```
---------------------------------------------------------------------
SUBJECT TO
 2       1       1.0000000      * X( 4)  **     1.0000000
                          .PLUS.
 2       2       1.0000000      * X( 5)  **     1.0000000
                          .PLUS.
 2       3       1.0000000      * X( 6)  **     1.0000000
                          .PLUS.
 2       4       1.0000000      * X( 7)  **     1.0000000
                                                          .LE.     1
---------------------------------------------------------------------
 3       1       0.18000000     * X( 1)  **    -1.0000000
                                * X( 2)  **     1.0000000
                                * X( 6)  **    -1.0000000
                                                          .LE.     1
---------------------------------------------------------------------
 4       1       40.000000      * X( 1)  **    -1.0000000
                                * X( 2)  **     1.0000000
                                * X( 3)  **    -1.0000000
                                * X( 5)  **    -1.0000000
                                                          .LE.     1
---------------------------------------------------------------------
```

```
---------------------------------------------------------------------

 5      1       44500000.       * X( 2) **      -1.0000000
                                * X( 4) **      -1.0000000

                        .PLUS.

 5      2       0.60000000D-07 * X( 1) **       1.0000000
                                * X( 2) **      -1.0000000
                                * X( 3) **       3.0000000
                                * X( 4) **      -1.0000000

                        .PLUS.

 5      3       0.21500000D-07 * X( 3) **       2.0000000
                                * X( 4) **      -1.0000000
                                * X( 7) **      -1.0000000

                                                        .LE.    1
---------------------------------------------------------------------


THE TOTAL NUMBER OF TERMS = 10
THE NUMBER OF 'DEGREES OF DIFFICULTY' = 2
```

DUAL PROGRAM.

THE NUMBER OF DUAL VARIABLES = 10
THE NUMBER OF DUAL EQUATIONS = 8

*** EXTENDED MATRIX OF COEFFICIENTS OF DUAL VARIABLES ***

1.000	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.000
0.0	1.000	0.0	0.0	0.0	0.0	0.0	0.0	-1.000	-1.000	1.000
0.0	0.0	1.000	0.0	0.0	0.0	0.0	0.0	-3.000	-2.000	0.0
0.0	0.0	0.0	1.000	0.0	0.0	0.0	0.0	2.000	2.000	1.000
0.0	0.0	0.0	0.0	1.000	0.0	0.0	0.0	0.0	-1.000	0.0
0.0	0.0	0.0	0.0	0.0	1.000	0.0	0.0	2.000	2.000	1.000
0.0	0.0	0.0	0.0	0.0	0.0	1.000	0.0	-3.000	-2.000	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.000	0.0	0.0	1.000

```
PATTERN SEARCH.
---------------


THE NUMBER OF VARIABLES =  2
THE NUMBER OF CONSTRAINTS = 14
THE LOWER LIMIT ON STEP SIZE =    1.0000D-06
PRINT OPTION =  2


INITIAL POINT.
OBJ.FCN.= 8.60619D 07 VARIAB.=  0.200000   0.200000
CONSTR.=   1.000000  1.400000  1.000000   0.200000   0.200000   0.200000   1.000000 1.
  0.200000  0.200000  1.000000  2.800000   0.200000   1.000000   1.400000
-------------------------------------------------------------

INCREMENTAL VALUE OF VARIABLES .LT. LOWER LIMIT.
PATTERN SEARCH TERMINATED.
OBJ.FCN.= 1.26748D 08 VARIAB.=  0.246988  0.042763
CONSTR.=   1.000000  1.289751  0.826490  0.420498  0.042763  0.420498   0.826490    1.
  0.246988  0.042763  1.000000  2.579502  0.420498  0.826490  1.289751
-------------------------------------------------------------
-------------------------------------------------------------
```

DUAL SOLUTION.

THE VALUE OF THE DUAL FUNCTION AT A MAXIMUM POINT =
0.12674758D 09

THE CORRESPONDING DUAL VARIABLES ARE.

DUAL VARIABLE		VALUE
w(1)	=	1.0000000
w(2)	=	1.2897512
w(3)	=	0.82649018
w(4)	=	0.42049763
w(5)	=	0.42763375D-01
w(6)	=	0.42049763
w(7)	=	0.82649018
w(8)	=	1.0000000
w(9)	=	0.24698781
w(10)	=	0.42763375D-01
w(11)	=	1.0000000
w(12)	=	2.5795024
w(13)	=	0.42049763
w(14)	=	0.82649018
w(15)	=	1.2897512

PRIMAL SOLUTION.

THE VALUES OF THE PRIMAL VARIABLES AT A MINIMUM POINT ARE

PRIMAL VARIABLE		VALUE
X(1)	=	0.12674758D 09
X(2)	=	0.11478755D 09
X(3)	=	113.06113
X(4)	=	0.50000000
X(5)	=	0.32040683
X(6)	=	0.16301502
X(7)	=	0.16578149D-01

THE CONTRIBUTION OF EACH TERM TO THE PRIMAL FUNCTION IS.

TERM	CONTRIBUTION
1	0.12674758D 09
THE VALUE OF ROW 1 =	0,12674758D 09
2	0.50000000
3	0.32040683
4	0.16301502
5	0.16578149D-01
THE VALUE OF ROW 2 =	1.0000000
6	1.0000000
THE VALUE OF ROW 3 =	1.0000000
7	1.0000000
THE VALUE OF ROW 4 =	1.0000000
8	0.77534539
9	0.19149884
10	0.33155768D-01
THE VALUE OF ROW 5 =	1.0000000

THE VALUE OF THE PRIMAL FUNCTION = 0.12674758D 09

THE VALUE OF THE DUAL FUNCTION = 0.12674758D 09

THE ABSOLUTE VALUE OF THE DIFFERENCE = 0.10952353D-05

THE RELATIVE DIFFERENCE = 0.109524D-05/ 0.126748D 09

= 0.000000 PER CENT

PRIMAL PROBLEM IS CONVEX. GLOBAL MINIMUM GUARANTEED.

ELAPSED TIME = 1.810 SECONDS.